Union of all the minimum cycle bases of a graph

Philippe VISMARA

L.I.R.M.M.
Université Montpellier II / CNRS
161 rue Ada,
34392 Montpellier Cedex 5, FRANCE
E-mail: vismara@lirmm.fr

E.N.S.A.-M.
2 Place Pierre Viala
34060 Montpellier Cedex 1, FRANCE
E-mail: vismara@ensam.inra.fr

Submited: October 10, 1996; Accepted: January 17, 1997.

Abstract

The perception of cyclic structures is a crucial step in the analysis of graphs. To describe the cycle vector space of a graph, a minimum cycle basis can be computed in polynomial time using an algorithm of [Horton, 1987]. But the set of cycles corresponding to a minimum basis is not always relevant for analyzing the cyclic structure of a graph. This restriction is due to the fact that a minimum cycle basis is generally not unique for a given graph. Therefore, the smallest canonical set of cycles which describes the cyclic structure of a graph is the *union of all the minimum cycle bases*. This set of cycles is called the *set of relevant cycles* and denoted by $\mathcal{C}_\mathcal{R}$. A relevant cycle can also be defined as a cycle which is not the sum of shorter cycles.

A polynomial algorithm is presented that computes a compact representation of the potentially exponential-sized set $\mathcal{C}_\mathcal{R}$ in $O(\nu m^3)$ (where ν denotes the cyclomatic number). This compact representation consists of a polynomial number of relevant cycle prototypes from which all the relevant cycles can be listed in $O(n\,|\mathcal{C}_\mathcal{R}|)$. A polynomial method is also given that computes the number of relevant cycles without listing all of them.

AMS Subject Classification: 05C85 (primary), 05C38 (secondary).

Introduction

For a graph $G = (V, E)$ with n vertices and m edges the set of elementary[1] cycles can be extended to form a vector space on $\{0, 1\}^m$. If G is connected, the dimension of this vector space is given by the *cyclomatic number* $\nu = m - n + 1$, initially introduced for polyhedrons by [Euler, 1752] and generalized by [Cauchy, 1813].

The simplest way to determine this vector space (denoted by $\mathcal{C}$) consists in finding a *fundamental basis associated with a spanning tree*[2] [Kirchhoff, 1847]. Many papers deal with fundamental basis search [Welch, 1966; Paton, 1969; Gibbs, 1969]. Most of them use this notion to list all elementary cycles of a graph. Unfortunately, as [Mateti and Deo, 1975] have shown, such a method can generate 2^ν vectors few of which actually correspond to elementary cycles. [Read and Tarjan, 1975] solve this problem using a search algorithm which needs $O(n + m + nc)$ operations, where c is the number of elementary cycles. For planar graphs, [Syslo, 1981] has shown that a vector space approach can be as efficient as a search algorithm.

Because fundamental cycle bases may be numerous, they are compared according to their lengths. The length of a cycle basis is the sum of the lengths of all cycles in the basis. This notion defines *minimum fundamental cycle bases*. [Deo *et al.*, 1982] have shown that finding a minimum fundamental cycle basis is an NP-complete problem. Furthermore, [Horton, 1987] has presented a polynomial time algorithm to find a minimum cycle basis not necessarily associated with a spanning tree.

Some applications require the perception of the cyclic parts of a graph. An obvious solution would be to list all elementary cycles. This method has two drawbacks: it cannot be performed efficiently and only a few of the elementary cycles may be relevant to the application domain. Another solution would be to find a minimum cycle basis, but such a set of cycles is generally not unique for a given graph.

Of course, the kind of perception which is to be performed determines the definition of an optimal set of cycles. In general, one requires a canonical set of cycles from which the cycle vector space can be generated. The smallest set satisfying this condition is the *union of all the minimum cycle bases*.

This paper is organized as follows: Section 2 presents Horton's algorithm to find a minimum cycle basis. Section 3 defines the *set of relevant cycles* (denoted by $\mathcal{C}_\mathcal{R}$) as the *union of all the minimum cycle bases*. We present a polynomial time algorithm to compute $\mathcal{C}_\mathcal{R}$ in terms of a polynomial number of cycle families. These define a partition of $\mathcal{C}_\mathcal{R}$. Each family is represented by a single cycle from which all the other cycles of the family can be directly generated. In the last section, we propose a polynomial method to calculate the number of relevant cycles (for the whole graph or including a given vertex).

[1] A cycle is called *elementary* if it contains no vertex more than once.

[2] Let $T = (V, E')$ be a spanning tree for the graph $G = (V, E)$. Adding any edge from $E \setminus E'$ to T creates a single cycle. The set of cycles generated this way defines a basis for the vector space $\mathcal{C}$.

1 Preliminary

In this paper, we consider an undirected 2-connected graph $G = (V, E)$ without loops or multiple edges. Considering 2-connected graphs only does not limit generality because cycle subspaces associated with 2-connected components are independent and complementary (their union is equal to the whole vector space). An edge is denoted by an unordered pair of vertices (x, y). A *cycle* is a subgraph such that any vertex degree is even. Therefore, we consider a connected cycle as either a set of edges $\{(x_1, x_2), (x_2, x_3), \dots\}$ or a closed path $(x_1, x_2, x_3, \dots, x_1)$. An *elementary* cycle is a minimal subgraph such that every node has degree 2.

Define the *sum* of two cycles $C + C'$ to be the symmetric set difference $(C \cup C') \setminus (C \cap C')$. Then, the cycle vector space is the additive closure of the set of elementary cycles. A cycle C can be represented by an element of the cycle vector space $\{0,1\}^m$ (i.e. $C[i] = 1 \Leftrightarrow$ *edge* i *belongs to* C).

For any vertex v in V, we denote by $deg(v)$ the degree of v in G and by $\Gamma(v)$ the set of vertices which are adjacent to v.

Each edge e is weighted with a real positive number $w(e)$. The weight of any path (or cycle) $\mu = (x_1, x_2, \dots, x_k)$ is defined by $w(\mu) = \sum_{i \in [1..k-1]} w(\,(x_i, x_{i+1})\,)$.

The *distance* $d(x, y)$ between vertices x and y is the weight of any shortest path from x to y.

2 Horton's algorithm

The algorithm proposed by [Horton, 1987] is the first polynomial algorithm that finds a minimum cycle basis. In fact, a quite similar method was presented by [Sorkau, 1985], but Sorkau did not analyze the complexity of his algorithm. Hence, we will only focus on Horton's algorithm which is based on the following theorem:

Theorem 1 *Let x be any vertex of any cycle C in a minimum cycle basis. There is an edge (y, z) in C such that C consists of a shortest path from x to y, a shortest path from x to z and the edge (y, z).*

Note that any cycle satisfying this property does not necessarily belong to a minimum basis. Therefore, Horton's algorithm extracts a cycle basis from an initial set of cycles (denoted by $\mathcal{C}_\mathcal{I}$) satisfying Theorem 1:

1) $\forall a, b \in V$ find a shortest path $P(a, b)$ between a and b.
2) For all $v \in V$ do:
    - For all $(x, y) \in E$ do:
        - If $P(v, x) \cap P(v, y) = \{v\}$
        - Then add to $\mathcal{C}_\mathcal{I}$ the cycle $C = P(v, x) + P(v, y) + (x, y)$
3) Order the initial set of cycles $\mathcal{C}_\mathcal{I}$ by weight
4) Use a greedy algorithm to extract a minimum cycle basis from $\mathcal{C}_\mathcal{I}$.

The first step of the algorithm chooses a unique shortest path for every pair of vertices. It can be performed in $O(n^3)$ using different algorithms [Floyd, 1962; Dijkstra, 1959].

Horton has proved that any choice can lead to an initial set $\mathcal{C}_\mathcal{I}$ including a minimum cycle basis. This proof consists in replacing the cycles of any minimum basis by cycles belonging to $\mathcal{C}_\mathcal{I}$. In the second stage, a cycle C may be created as many times as it contains vertices. To avoid this duplication, Horton uses the method suggested by [Tierman, 1970]. Considering an order π on vertices, all cycles generated from vertex v must only contain vertices that precede v in order π.

Given a vertex v and an edge (x, y), testing "$P(v, x) \cap P(v, y) = \{v\}$" is necessary to check if the corresponding cycle C is *degenerate* (i.e. at least one vertex of the associated subgraph has a degree equal to 1 or greater than 2). This test needs $O(n)$ operations. So, the second step of Horton's algorithm takes $O(n^2\, m)$ operations.

The last step of the algorithm processes $\mathcal{C}_\mathcal{I}$ in order of the weight of the cycles. A new cycle is added to the k cycles of the current partial basis when the new induced set is independent. This test uses a Gaussian elimination on the $m \times (k+1)$ boolean matrix corresponding to the set of cycles (each row is a vector of $\{0, 1\}^m$, representing a cycle). The elimination can be performed in $O(m \times (k+1))$. Since $k \leq \nu$ and $\mathcal{C}_\mathcal{I}$ contains at most νn cycles[3], a minimum cycle basis can be found in $O(m\nu^2 n)$ operations.

3 Union of the all minimum cycle bases

A minimum cycle basis is a compact representation of the cycle vector space of a graph. However, it does not necessarily include all cycles relevant to a given problem.

In Organic Chemistry, the constitution of a chemical compound can be viewed as a labeled graph. The perception of cyclic parts is of fundamental importance in the analysis of molecules. Most of the programs dealing with chemistry require a fast and accurate method for the identification of the "chemically meaningful" cycles among the potentially large number of elementary cycles embedded in the molecular graph. The most commonly used restriction is a minimum cycle basis. Indeed, for a cycle to be *chemically relevant*, it must not be the sum of smaller cycles. However, in general, a given graph has several minimum bases. Therefore the above definition of chemically relevant cycles is not satisfactory. For example, the graph presented in Figure 1 has three different minimum bases.

The smallest generating set of cycles which is unique for a given graph is the ***union of all the minimum bases*** of the graph.

[3]In step 2, a cycle is added to $\mathcal{C}_\mathcal{I}$ when it consists of two distinct shortest paths. Hence, the number of cycles created from a vertex v is at most equal to the number of cycle closure edges.

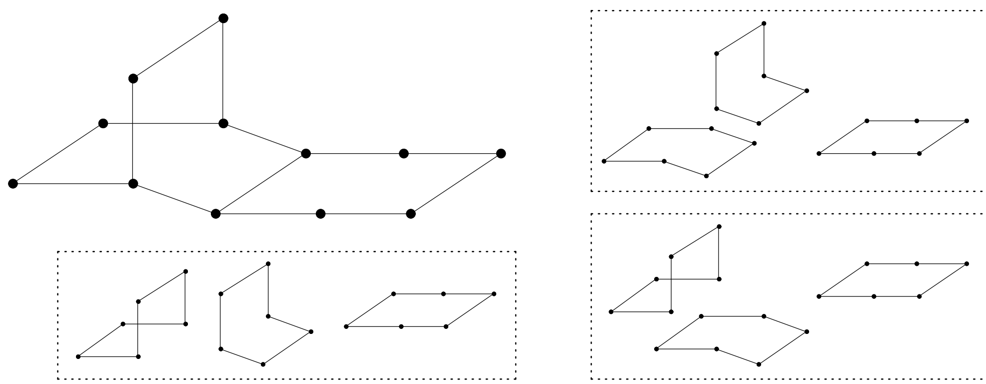


Figure 1: An example of a graph having several minimum bases.

Definition 1 *A cycle is said to be* ***relevant*** *if it belongs to at least one minimum cycle basis.*

So, the *set of relevant cycles*, denoted by $\mathcal{C}_\mathcal{R}$, is the union of all the minimum cycle bases of the graph.

Another characterization of relevant cycles is given by the following lemma:

Lemma 1 $C \in \mathcal{C}$ *is* relevant $\Leftrightarrow$ *no elementary cycles* $C_1, \ldots, C_k$ *exist such that* $C = C_1 + \ldots + C_k$ *and* $\forall i \in [1..k],\ w(C_i) < w(C)$.

Proof *If* C *belongs to a minimum cycle basis then it cannot be the sum of shorter cycles. Otherwise,* C *could be exchanged by one of these cycles in the minimum basis.*

Conversely, let $\mathcal{B}$ *be a minimum cycle basis and* C *a cycle which is not the sum of shorter cycles. Then* $C = B_1 + \ldots + B_k$*, where* $\forall i, B_i \in \mathcal{B}$*, and there is at least one cycle* B_i *such that* $w(B_i) \geq w(C)$*. Since* $\mathcal{B}$ *is a minimum basis, we have* $w(B_i) = w(C)$*. So, the set* $(\mathcal{B} \setminus \{B_i\}) \cup \{C\}$ *is a minimum basis* $\square$

This definition of cycle relevance has been suggested by [Plotkin, 1971] in the domain of computational chemistry. Since no efficient algorithm was proposed to find these cycles, many works deal with the definition of extended minimum cycle bases which are generally not canonical (see [Downs *et al.*, 1989]).

The cardinality of the union of the minimum cycle bases is not necessarily polynomial. If G is a complete graph, the cardinality of $\mathcal{C}_\mathcal{R}$ is equal to the polynomial number of 3-edge cycles. But some graphs can include an exponential number of *relevant* cycles, such as the one given in Figure 2.

The existence of such graphs must not hide the interest of the *relevance* notion. Firstly, because in some domains, such as organic chemistry, graphs with an exponential number of *relevant* cycles are almost impossible. Secondly, we present a polynomial time algorithm to compute $\mathcal{C}_\mathcal{R}$ as a polynomial number of parts. Each

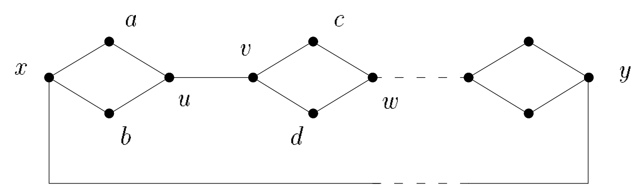


Figure 2: *The union of all the minimum cycle bases of this graph includes* $2^{\frac{n}{4}}$ *cycles with* $\frac{3n}{4}$ *vertices (* $(x, b, u, v, d, w, \ldots, y, x)$, $(x, a, u, v, d, w, \ldots, y, x)$, *etc.).*

part corresponds to a *family* of cycles which can be directly listed from a particular relevant cycle. This cycle defines a *prototype* of its family.

Representing $\mathcal{C}_{\mathcal{R}}$ as a polynomial number of cycle prototypes provides a way of determining the cardinality of $\mathcal{C}_{\mathcal{R}}$ in polynomial time. In section 7.2, we present a method to find the number of cycles which can be generated from a given prototype, without calculating any of them. This compact representation of $\mathcal{C}_{\mathcal{R}}$ gives a more precise description of the cycle vector space than a simple minimum basis. For instance, in the graph of Figure 2, apart from the $\frac{n}{4}$ 4-edge cycles that must be known, the other *relevant* cycles can be described by a single family. The prototype of this family is one of the $2^{\frac{n}{4}}$ elementary cycles with $\frac{3n}{4}$ vertices. Listing all these *relevant* cycles is not very meaningful for such a graph. But it can be very useful for a computer system to be able to perceive them and designate a single prototype.

Of course, the compact representation can always be replaced by a complete enumeration of the relevant cycles. It provides a canonical set of shortest cycles from which the whole vector space can be generated.

4 Partition of the set of relevant cycles $\mathcal{C}_{\mathcal{R}}$

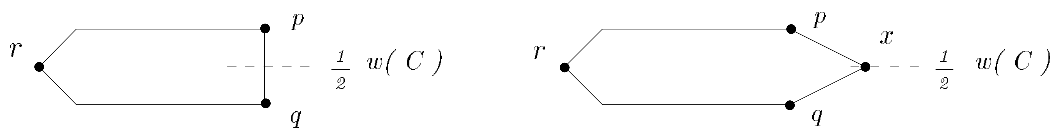
This section defines a partition of the set of relevant cycles such that the number of parts is polynomial.

Lemma 2 *If* μ *is a subpath of a relevant cycle* C *such that* $w(\mu) \leq \frac{1}{2}w(C)$ *then* μ *is a shortest path.*

> **Proof** *Since* C *is relevant, it belongs to at least one minimum cycle basis* $\mathcal{B}$. *Assume that* μ *is not a shortest path. Then* μ *includes a subpath* $\rho = (a \ldots b)$ *such that* $w(\rho) > d(a, b)$ *and there is a shortest path* ρ' *from* a *to* b *such that* ρ *and* ρ' *are disjoint. Replacing* ρ *with* ρ' *in* C *will create a new cycle* C' *such that* $w(C') < w(C)$. *Define* C'' *to be the cycle that consists of* ρ *and* ρ'. *Since* $w(\rho') < w(\rho) \leq \frac{1}{2}w(C)$, $w(C'') < w(C)$. *Because* $C = C' + C''$, C *can be exchanged for either* C' *or* C'' *in* $\mathcal{B}$. *So* $\mathcal{B}$ *cannot be a minimum basis* $\Box$

In the rest of the paper, we assume that the vertices of the graph being considered are ordered by a numbering π. This ordering can be chosen arbitrarily. For any cycle C being considered, we denote by r the greatest vertex in C according to π.

Definition 2 *any cycle C is "**even**" if it consists of two disjoint paths μ_1 and μ_2 from r to x, such that r is the greatest vertex in C (i.e. $\pi(r) = \max_{z \in C} \pi(z)$) and $w(\mu_1) = w(\mu_2) = \frac{1}{2}w(C)$. Otherwise, C is "**odd**".*



Odd cycle *Even cycle*

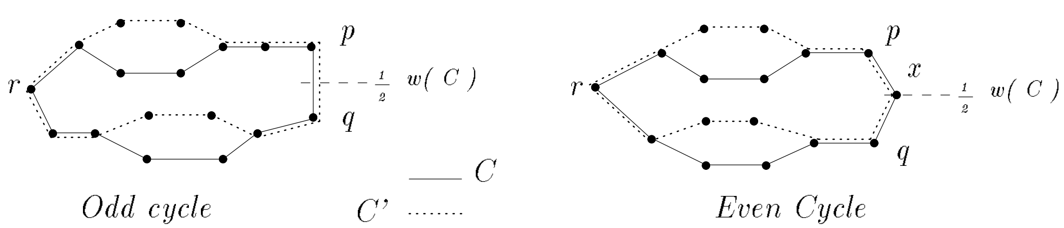
Theorem 2 *any relevant cycle C consists of two disjoint shortest paths $(r \dots p)$ and $(r \dots q)$ linked by the edge (p, q) if C is* odd *or by the path (p, x, q) if C is* even.

Proof *If C is even, C consists of two disjoint paths $(r, \dots, p, x)$ and $(r, \dots, q, x)$ whose weight is $\frac{1}{2}w(C)$. Hence, $w(\,(r, \dots p)\,) \leq \frac{1}{2}w(C)$ and $w(\,(r, \dots q)\,) \leq \frac{1}{2}w(C)$, because the valuation is positive. If C is odd, $C = (r, \dots p, q, \dots, r)$ where $w(\,(r, \dots p)\,) < \frac{1}{2}w(C)$ and $w(\,(q, \dots r)\,) < \frac{1}{2}w(C)$.*
By lemma 2, $(r, \dots p)$ and $(r, \dots q)$ are shortest paths in both cases □

The partition of the set of relevant cycles is based on theorem 2. For any relevant cycle C including the vertices r, p, q and eventually x, as defined in theorem 2, we define the *cycle family* associated with C as follows :

Definition 3

$$\mathcal{F}(C) = \left\{ C' \in \mathcal{C}_{\mathcal{R}} \;\middle|\; \begin{array}{l} w(C') = w(C) \text{ and } C' \text{ consists of :} \\ \bullet \text{ the vertex } r \text{ and the edge } (p,q) \text{ (or the path } (p,x,q)), \\ \bullet \text{ two shortest paths } (r, \dots, p) \text{ and } (r, \dots, q) \text{ passing} \\ \quad \text{only through vertices smaller than } r. \end{array} \right\}$$



Odd cycle *Even Cycle*

Hence, two cycles C and C' belonging to $\mathcal{F}(C)$ only differ on the shortest paths from r to p and from r to q that they include.

Theorem 3 *The set of all the relevant cycle families defines a partition of $\mathcal{C}_{\mathcal{R}}$.*

Proof *By theorem 2 and definition 3, the cycle family associated with a relevant cycle C is unique for a given ordering π. So any relevant cycle belongs to exactly one family. Then, we define an equivalence relation such that two relevant cycles are equivalent if one belongs to the cycle family of the other* □

To describe the family $\mathcal{F}(C)$, we need a single cycle C which is a *prototype* of this family. All the other cycles in $\mathcal{F}(C)$ can be generated from C by replacing paths $(r, \dots p)$ and $(r, \dots q)$ with paths of the same weight passing only through vertices smaller than r.

Hence, a cycle family is properly defined by a triple r, p, q for odd cycles or a quadruple r, p, q, x for even cycles.

The number of relevant cycle families depends on the order π used to define the families. However, we have the following result :

Theorem 4 *the number of relevant cycle families is always polynomial.*

> **Proof** *The number of families whose prototype is an odd cycle is smaller than $n\,\nu$, since an odd cycle prototype is defined by a vertex r and a cycle closing edge (p, q).*
> *As for even cycle families, their prototype is defined by 4 vertices r, x, p and q such that p and q are adjacent to x. Hence, the number of even cycle families is smaller than $\sum_{r \in V} \sum_{x \in V, \pi(x) \le \pi(r)} \frac{1}{2} deg(x)^2 \;\le\; \frac{1}{2} n^2 m$.*
> *If the order π respects vertex degree $(\forall x \in V,\ \pi(x) \le \pi(v) \Rightarrow deg(x) \le deg(r))$ there are at most $\sum_{r \in V} \left(deg(r) \times \sum_{x \in V, \pi(x) \le \pi(r)} \frac{1}{2} deg(x) \right) \le 2m^2$ even cycle families* $\square$

To compute the union of all the minimum cycle bases, we may find one prototype for each relevant cycle family.

5 Computation of cycle prototypes

The algorithm we propose to compute cycle prototypes is based on the converse of theorem 2. This converse is not necessarily true but it gives a strong condition on cycle relevance.

The outline of the algorithm can be given as follows:

- firstly, we compute the set $\mathcal{C}'_{\mathcal{I}}$ including one cycle for each triple r, p, q (or quadruple r, p, q, x) satisfying the condition of theorem 2,
- secondly, we use a greedy algorithm to extract relevant cycles from $\mathcal{C}'_{\mathcal{I}}$.

5.1 Computation of the set $\mathcal{C}'_{\mathcal{I}}$

Before we give a more precise description of the first step of the algorithm, we need to introduce a new definition :

Definition 4 *For any vertex r,*

$$V_r = \left\{ z \in V \text{ such that } \middle| \begin{array}{l} \textit{there is a shortest path in } G \textit{ from } r \textit{ to } z \textit{ that passes only} \\ \textit{through vertices which precede } r \textit{ in the ordering } \pi. \end{array} \right\}$$

Then, we can give the outline of the algorithm that computes $\mathcal{C}'_{\mathcal{I}}$:

[Algorithm 1]

1. For all $r \in V$ do:
2. Compute V_r and $\forall t \in V_r$ find a shortest path $P(r,t)$ from r to t;
3. For all $y \in V_r$ do:
4. $\mathcal{S} \leftarrow \varnothing$;
5. For all $z \in V_r$ such that z is adjacent to y do:
6. If $d(r,z) + w(\,(z,y)\,) = d(r,y)$ Then
7. $\mathcal{S} \leftarrow \mathcal{S} \cup \{z\}$; $\star$ *z belongs to a shortest path from r to y* $\star$
8. Else If $\begin{pmatrix} d(r,z) \neq d(r,y) + w(\,(z,y)\,) \text{ and } \pi(z) < \pi(y) \\ \text{and } P(r,y) \cap P(r,z) = \{r\} \end{pmatrix}$ Then
9. Add to $\mathcal{C}'_{\mathcal{I}}$ the odd cycle $C = P(r,y) + P(r,z) + (z,y)$;
10. EndIf
11. EndFor
12. For any pair of vertices p,q in $\mathcal{S}$ such that $P(r,p) \cap P(r,q) = \{r\}$ Do:
13. *Add to* $\mathcal{C}'_{\mathcal{I}}$ *the even cycle* $C = P(r,p) + P(r,q) + (p,y,q)$;
14. EndFor
15. EndFor
16. EndFor

For a given vertex r, the set V_r can be computed at line 2 using a shortest path algorithm. For example, the algorithm presented in [Dijkstra, 1959] can be easily updated to compute V_r by preferably choosing the shortest paths which pass only through vertices smaller than r. This variant does not change the algorithm complexity. It remains in $O(n^2)$ operations when the priority queue is implemented as an array, $O(m \log n)$ using a binary heap or $O(m + n \log n)$ with a Fibonacci heap.

To generate the prototypes associated with a vertex r, we explore any vertex y in V_r. We are now going to detail the generation of $\mathcal{C}'_{\mathcal{I}}$ illustrated by Figure 3.

Even cycles generated for y :

$(r, \ldots, z_1, y, z_2, \ldots, r)$
$(r, \ldots, z_1, y, z_3, \ldots, r)$
$(r, \ldots, z_2, y, z_3, \ldots, r)$

Odd cycle generated for y :

$(r, \ldots, z_1, y, z_4, \ldots, r)$
where $(r, \ldots, z_1) = P(r,y)$

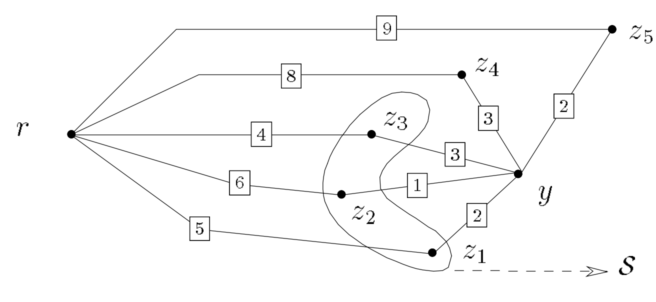


Figure 3: Generation of $\mathcal{C}'_{\mathcal{I}}$.

First, we compute the set $\mathcal{S} = \{\, z \in \Gamma(y) \cap V_r \mid d(r,z) + w((z,y)) = d(r,y) \,\}$. So, at the end of line 11, $\mathcal{S}$ includes all the vertices in V_r adjacent to y such that there is a shortest path from r to y passing through the edge (z,y). For any pair of vertices p,q in $\mathcal{S}$, we create the even cycle $C = P(r,p) + P(r,q) + (p,y,q)$ when paths $P(r,p)$ and $P(r,q)$ are disjoint (see line 13).

As for odd cycles, they are created at line 9 for any vertex z in $\Gamma(y) \cap V_r$ such that $d(r,y) - w(\,(z,y)\,) < d(r,z) < d(r,y) + w(\,(z,y)\,)$. To avoid cycle duplication, we only consider any vertex z such that $\pi(z) < \pi(y)$.

A quick analysis of algorithm 1 gives for each vertex r in V:

- line 2 takes $O(m + n \log n)$ operations,
- for lines 5 to 11, the critical step is line 8 which takes $\sum_{y \in V_r} O(deg(y)) \times O(n) \;=\; O(n\,m)$ operations, where $O(n)$ is necessary to check if the paths are disjoint,
- line 12-14 takes $\sum_{y \in V_r} O(deg(y))^2 \times O(n) \;\leq\; O(m\,n\,deg(r))$ if we suppose that the order π respects the vertex degree (i.e. $\forall y \in V_r,\; deg(y) \leq deg(r)$).

So, algorithm 1 takes $\boldsymbol{O(n\,m^2)}$ operations.

Lemma 3 *If order π satisfies $\forall x, y \in V,\quad \pi(x) \leq \pi(y) \Rightarrow deg(x) \leq deg(y)$ then the cardinality of set $\mathcal{C}'_{\mathcal{I}}$ is lower than $(2m^2 + \nu n)$.*

Proof *The number of* odd *cycles in $\mathcal{C}'_{\mathcal{I}}$ is less than νn. As for the* even *cycles, they are at most $\sum_{r \in V} \sum_{y \in V_r} \frac{1}{2}\mathrm{deg}(y)^2$. For any y in V_r, $\mathrm{deg}(y) \leq deg(r)$. So, there are at most $\sum_{r \in V} \left(\mathrm{deg}(r) \times \sum_{y \in V_r} \frac{1}{2}\mathrm{deg}(y)\right) \leq 2m^2$ even cycles in $\mathcal{C}'_{\mathcal{I}}$* $\square$

Theorem 5 *$\mathcal{C}'_{\mathcal{I}}$ includes one and only one prototype of each relevant cycle family.*

Proof *Let $C = (r, \ldots, p, x, q, \ldots, r)$ be a relevant even cycle (if C is odd, the proof is quite similar). Assume that r is the greatest vertex in C according to π and x is the unique vertex in C such that $d(r,x) = \frac{1}{2}w(C)$.*
Now consider the exploration of vertex $y = x$ in algorithm 1. By theorem 2, $(r, \ldots, p)$ and $(r, \ldots, q)$ are shortest paths. So p and q belong to $\mathcal{S}$. Define C' to be the cycle added to $\mathcal{C}'_{\mathcal{I}}$ at line 13 for the pair of vertices p, q. By construction of $\mathcal{C}'_{\mathcal{I}}$, C' is unique.
Cycles C and C' only differ by the shortest paths $(r, \ldots, p)$ and $(r, \ldots, q)$ that they include. The sum $C + C'$ defines a set of cycles which are smaller than C. Hence, cycle C' must be relevant since C is relevant. So, C' is the unique prototype of $\mathcal{F}(C)$ which belongs to $\mathcal{C}'_{\mathcal{I}}$ $\square$

5.2 Computation of a set of prototypes

By theorem 5, $\mathcal{C}_{\mathcal{R}} \cap \mathcal{C}'_{\mathcal{I}}$ is a set of prototypes. So, we have to extract all the relevant cycles in $\mathcal{C}'_{\mathcal{I}}$.

As in Horton's algorithm, a minimum cycle basis $\mathcal{B}$ is calculated by processing the cycles of $\mathcal{C}'_{\mathcal{I}}$ in order of their weight. During the processing of a cycle C, we denote by $\mathcal{B}_<$ and $\mathcal{B}_=$ the subsets of cycles in the current sub-basis whose weights are less than $w(C)$ and equal to $w(C)$, respectively. By Lemma 1, C is relevant when the set

$\mathcal{B}_< \cup \{C\}$ is independent. When this check succeeds, cycle C is added to $\mathcal{B}_=$ if the set $\mathcal{B}_< \cup \{C\} \cup \mathcal{B}_=$ is also independent. The processing stops when $\mathcal{B}$ is complete and all the cycles having the same weight as the greatest one in $\mathcal{B}$ have been tested.

The current minimal sub-basis of $\mathcal{B}$ is represented by a $(|\mathcal{B}_<| + |\mathcal{B}_=|) \times m$ boolean matrix. To test if set $\mathcal{B}_< \cup \{C\}$ is independent, we perform a Gaussian elimination on the first rows of the matrix which are associated with $\mathcal{B}_<$. When this check succeeds, the Gaussian elimination follows on the other rows to test if set $\mathcal{B}_< \cup \mathcal{B}_= \cup \{C\}$ is also independent.

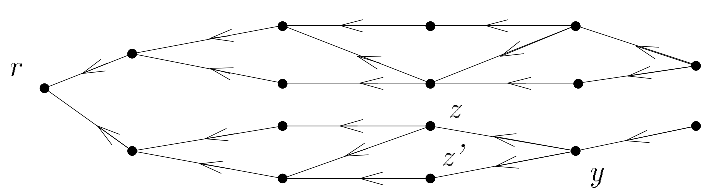
Since $\mathcal{B}$ contains at most ν cycles, each check of a cycle is performed in $O(\nu m)$. By lemma 3, the cardinality of $\mathcal{C}'_{\mathcal{I}}$ is in $O(m^2)$. So, the set of prototypes $\mathcal{C}'_{\mathcal{I}} \cap \mathcal{C}_{\mathcal{R}}$ can be generated in $\boldsymbol{O(\nu\, m^3)}$ operations.

6 Enumeration of $\mathcal{C}_{\mathcal{R}}$

$\mathcal{C}_{\mathcal{R}}$ is completely described by a set of prototypes of the families defined in section 4. From this set of prototypes, the union of all the minimum cycle bases can easily be listed.

Consider a cycle prototype $C \in \mathcal{C}_{\mathcal{R}} \cap \mathcal{C}'_{\mathcal{I}}$. To generate the cycle family $\mathcal{F}(C)$, we define a directed graph $D_r = (V_r, U_r)$ associated with the vertex r such that:

$$U_r = \left\{ \textit{directed edge } (y,z) \textit{ such that } \;\middle|\; \begin{array}{l} (z,y) \text{ belongs to a shortest path in } G \text{ from } r \text{ to } y \\ \text{that passes only through vertices which precede} \\ r \text{ in the ordering } \pi. \end{array} \right\}$$



The computation of the digraph D_r can be performed directly in algorithm 1. For each directed edge (y, z) in U_r, z belongs to the set $\mathcal{S}$ which is computed at line 7 of the algorithm.

The digraph D_r has two major properties:

- It has no directed cycle,
- $\forall x \in V_r$, any directed path $(x, \ldots, r)$ in D_r corresponds to a shortest path in G.

To list all the paths from x to r in D_r, we use a backtracking function which is based on a deep first search from x. This recursive function can be resumed as follows:

```
Function List_Paths( x; current_path )
    add x at the head-end of current_path;
    If x = r Then Return({current_path})
    Else:
        Result ← ∅;
        For any z such that (x, z) ∈ U_r Do
            Result ← Result ∪ List_Paths(z, current_path);
        Return(Result)
```

To compute $\mathcal{F}(C)$, we first replace the path $(p, \ldots, r)$ in C by each one of the paths returned by the call List_Paths(p, $\emptyset$). Then, in any cycle generated this way, we replace the path $(q, \ldots, r)$ by each one of the paths resulting from List_Paths(q, $\emptyset$).

Each cycle in $\mathcal{F}(C)$ corresponds to a pair of paths $(p, \ldots, r)$, $(q, \ldots, r)$. Since the computation of each path takes a number of operations in order of the path cardinality, the family $\mathcal{F}(C)$ can be listed in $O(\, n\,|\mathcal{F}(C)|\,)$.

So, the generation of $\mathcal{C}_{\mathcal{R}}$ from subset $\mathcal{C}_{\mathcal{R}} \cap \mathcal{C}'_{\mathcal{I}}$ is performed in $O(n\,|\mathcal{C}_{\mathcal{R}}|)$ operations (including cycle enumeration).

Note that the use of prototypes substantially optimizes the generation of $\mathcal{C}_{\mathcal{R}}$: if a cycle prototype is relevant, this implies that all cycles of the corresponding family are relevant. Therefore, the number of cycle relevance checks is polynomial since it is in $O(|\mathcal{C}'_{\mathcal{I}}|)$ instead of $O(|\mathcal{C}_{\mathcal{R}}|)$.

7 Computation of the number of relevant cycles

7.1 Size of $\mathcal{C}_{\mathcal{R}}$

Using cycle families to describe $\mathcal{C}_{\mathcal{R}}$ provides a polynomial time algorithm to compute the size of this set without generating all cycles. This is particularly interesting for real-world problems to determine whether the number of relevant cycles is polynomial or not.

To perform this computation we may evaluate the number of shortest paths from r to any vertex x, which passe only throught vertices in V_r. This number is denoted by $\psi_r(x)$. Of course, $\psi_r(r) = 1$.

In algorithm 1, we compute, for each vertex y in V_r, the set $\mathcal{S}$ of all the vertices z in V_r such that there is a shortest path from r to y which ends in the edge (z, y).

Then, one can easily prove that: $\forall y \in V_r, \quad \psi_r(y) = \sum_{z \in \mathcal{S}} \psi_r(z)$

This result gives an efficient method to compute the function ψ_r if the vertices in V_r are processed according to a *topological sort*[4] of the graph D_r.

[4]In a graph without directed cycles, a topological sort is an ordering on the vertices such that : if there is a directed edge from x to y then x is greater than y.

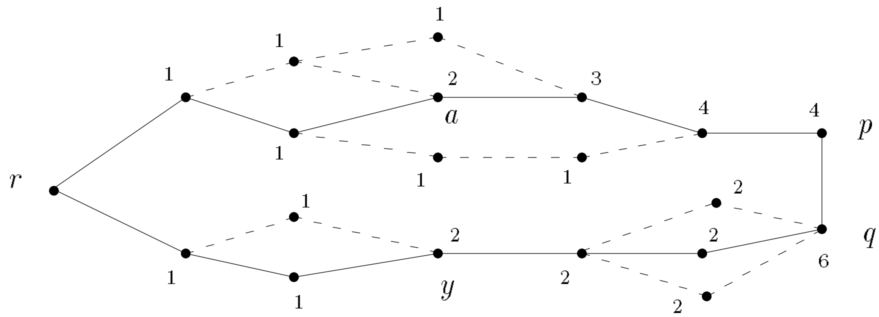


Figure 4: Since $\psi_r(p) = 4$ and $\psi_r(q) = 6$, cycle family $\mathcal{F}(C^r_{p,q})$ (corresponding to cycle $C^r_{p,q}$ in solid lines), contains 24 relevant cycles.

Denote by $C^r_{p,q}$ a cycle created from vertices r, p and q (and eventually x) according to the previous notations.

If $C^r_{p,q}$ belongs to $\mathcal{C}_\mathcal{R} \cap \mathcal{C}'_\mathcal{I}$, then the number of relevant cycles in family $\mathcal{F}(C^r_{p,q})$ is equal to $\psi_r(p) \times \psi_r(q)$ (see Figure 4). Consequently,

$$|\mathcal{C}_\mathcal{R}| = \sum_{C^r_{p,q} \in \mathcal{C}_\mathcal{R} \cap \mathcal{C}'_\mathcal{I}} \psi_r(p) \times \psi_r(q)$$

7.2 Number of relevant cycles including a given vertex

Determination of the union of all the minimum cycle bases as a polynomial number of cycle prototypes has still another interest. It provides a way of calculating, in polynomial time, the number of relevant cycles of a given weight which include each vertex. For example, in Figure 4, vertex a belongs to two 4-edge relevant cycles, one 6-edge cycle and twelve 13-edge ones. This knowledge can be useful to distinguish vertices according to their membership to more or less complex cyclic structures. For example, in organic chemistry this can help for taxonomy purposes in the determination of nomenclature names.

Let us consider a cycle family $\mathcal{F}(C^r_{p,q})$. Denote by $\#in(y, \mathcal{F}(C^r_{p,q}))$ the number of cycles in $\mathcal{F}(C^r_{p,q})$ which include vertex y. Assume that vertex y belongs to path $(r, .., q)$. Define $\phi_r(y, q)$ to be the number of shortest paths from y to q which are included in D_r. For example, in Figure 4, $\phi_r(y, q) = 3$.

We can determine $\phi_r(y, q)$ using a topological sort, in the same way as for the determination of ψ_r. This search needs $O(m)$ operations. Since there are at most $O(m^2)$ families, the global complexity is $O(m^3)$.

Then, one can easily prove the following result:

$$\#in(y, \mathcal{F}(C^r_{p,q})) = \psi_r(y) \times \phi_r(y, q) \times \psi_r(p)$$

In Figure 4, for vertex y, $\#in(y, \mathcal{F}(C^r_{p,q})) = 2 \times 3 \times 4 = 24$.

Conclusion

This paper deals with the computation of the *union of all the minimum cycle bases* of a graph. We present the first efficient algorithm to find this set of cycles. We call the *union of all the minimum cycle bases* the *set of relevant cycles* and denote it by $\mathcal{C}_\mathcal{R}$. It is the smallest generating set of the cycle space which is unique for a given graph. We propose a polynomial time algorithm to compute $\mathcal{C}_\mathcal{R}$ as a set of cycle families. Each family is represented by a cycle prototype and all the families constitute a partition of $\mathcal{C}_\mathcal{R}$. From the set of prototypes, all the relevant cycles can be listed in $O(n|\mathcal{C}_\mathcal{R}|)$ time (including output). We also propose a polynomial method to determine the number of relevant cycles which include a given vertex.

The use of prototypes provides an efficient compact representation of the union of all the minimum cycle bases. For real-world applications, the enumeration of $\mathcal{C}_\mathcal{R}$ is generally unnecessary.

One can envisage two major improvements for the problem of the union of all the minimum cycle bases. Firstly, it would be interesting to find an ordering on the vertices such that the number of relevant cycle families is minimum. Secondly, the algorithm we propose is polynomial but it can probably be improved. The critical step is the extraction of relevant cycles from the initial set $\mathcal{C}'_\mathcal{I}$. If a strong condition were found for characterizing relevant cycles without a check for independence, the improvement could be very significant. This is also true in the case of Horton's algorithm to compute a minimum cycle basis. It seems to be that an improvement to Horton's algorithm would entail an improvement to the one proposed here.

An other open problem is the enumeration of all the minimum cycle bases of a graph. It would be interesting to find a polynomial way to represent all these bases. But this problem has probably no real application.

Conversely, the notion of relevant cycles can be applied to real-world problems. The union of all the minimum cycle bases has been successfully used in the system RESYN [Vismara *et al.*, 1992; Vismara, 1995], a program for the planning of complex organic syntheses, to evaluate the set of chemically relevant cycles in a molecule.

References

[Cauchy, 1813] A. L. Cauchy. Recherche sur les polyèdres. *J. Ecole Polytechnique*, 9(16):68–86, 1813.

[Deo *et al.*, 1982] N. Deo, G. M. Prabhu, and M. S. Krishnamoorthy. Algorithms for generating fundamental cycles in a graph. *ACM Trans. Math. Software*, 8:26–42, 1982.

[Dijkstra, 1959] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[Downs *et al.*, 1989] G. M. Downs, V. J. Gillet, J. D. Holliday, and M. F. Lynch. Review of ring perception algorithms for chemical graphs. *J. Chem. Inf. Comput. Sci.*, 29(3):172–187, 1989.

[Euler, 1752] L. Euler. Elementa doctrinae solidorum. *Novi. Comm. Acad. Sci. Imp. Petropol.*, 4:109–140, 1752.

[Floyd, 1962] R.W. Floyd. Algorithm 97: shortest path. *Comm. ACM*, 5(6):345, 1962.

[Gibbs, 1969] N. Gibbs. A cycle generation algorithm for finite undirected linear graphs. *J. ACM*, 16:564–568, 1969.

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

[Kirchhoff, 1847] G. Kirchhoff. Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme geführt wird. *Poggendorf Ann. Physik*, 72:497–508, 1847. English translation in *Trans. Inst. Radio Engrs.*, CT-5:4–7, 1958.

[Mateti and Deo, 1975] P. Mateti and N. Deo. On algorithms for enumerating all circuits of a graph. *SIAM J. Comput.*, 5:90–101, 1975.

[Paton, 1969] K. Paton. An algorithm for finding a fundamental set of cycles of a graph. *Comm. ACM*, 12:514–518, 1969.

[Plotkin, 1971] M. Plotkin. Mathematical basis of ring-finding algorithms in CIDS. *J. Chem. Doc.*, 11:60–63, 1971.

[Read and Tarjan, 1975] R. C. Read and R. E. Tarjan. Bounds on backtrack algorithms for listing cycles, paths and spanning trees. *Networks*, 5:237–252, 1975.

[Sorkau, 1985] E. Sorkau. Ringerkennung in chemischen Strukturen mit dem Computer. *Wiss. Z. Tech. Hochsch. Leuna-Meuseburg*, 27:765–770, 1985.

[Syslo, 1981] A. A. Syslo. An efficient cycle vector space algorithm for listing all cycles of a planar graph. *SIAM Journal on Computing*, 10:797–808, 1981.

[Tierman, 1970] J. Tierman. An efficient search algorithm to find the elementary circuits of a graph. *Comm. ACM*, 13:722–726, 1970.

[Vismara *et al.*, 1992] P. Vismara, J-C. Régin, J. Quinqueton, M. Py, C. Laurenço, and L. Lapied. RESYN : Un système d'aide à la conception de plans de synthèse en chimie organique. In *Proceedings 12th International Conference Avignon'92*, volume 1, pages 305–318, Avignon, 1992. EC2.

[Vismara, 1995] P. Vismara. *Reconnaissance et représentation d'éléments structuraux pour la description d'objets complexes. Application à l'élaboration de stratégies de synthèse en chimie organique.* Thèse de doctorat, Université Montpellier II, 1995.

[Welch, 1966] J. Welch. A mechanical analysis of the cyclic structure of undirected linear graphs. *J. ACM*, 13:205–210, 1966.