Mathematical Models and a Constructive Heuristic for finding Minimum Fundamental Cycle Bases

LEO LIBERTI, EDOARDO AMALDI, NELSON MACULAN[1], FRANCESCO MAFFIOLI

DEI, Politecnico di Milano, P.zza L. da Vinci 32, 20133 Milano, Italy

({`liberti,amaldi,maculan,maffioli`}`@elet.polimi.it`)

3 December 2003

Abstract

The problem of finding a fundamental cycle basis with minimum total cost in a graph arises in many application fields. In this paper we present some integer linear programming formulations and we compare their performances, in terms of: instance size, CPU time required for the solution, and quality of the associated lower bound derived by solving the corresponding continuous relaxations. Since only very small instances can be solved to optimality with these formulations and very large instances occur in a number of applications, we present a new constructive heuristic and compare it with other existing heuristics.
Keywords: fundamental cycle, cycle basis, IP formulation, tree-growing heuristic.

1 Introduction

Let $G = (V, E)$ be a simple, undirected, biconnected graph with n nodes and m edges, where each edge $e \in E$ is assigned a weight $w_e \in \mathbb{R}$. A *cycle* is a subset C of E such that every node of V is incident with an even number of edges in C. Since an elementary cycle is a connected cycle such that at most two edges are incident to any node, cycles can be viewed as the (possibly empty) union of edge-disjoint elementary cycles. If cycles are considered as edge-incidence binary vectors in $\{0,1\}^{|E|}$, it is well-known that the cycles of a graph form a vector space over $GF(2)$. A set of cycles is a *cycle basis* if it is a basis in this cycle vector space associated to G. The cost of a cycle is the sum of the costs of all edges contained in the cycle. The cost of a set of cycles is the sum of the costs of all cycles in the set. Given any spanning tree T of G, the edges in $G \backslash T$ (the co-tree) are called *chords* of G with respect to T. Any chord uniquely identifies a cycle consisting of the chord itself and the unique path in T connecting the two vertices incident on the chord. These $m - n + 1$ cycles are called *fundamental cycles*; they form a *Fundamental Cycle Basis* (FCB) of G with respect to T. It turns out [Sys81] that a cycle basis is fundamental if and only if each cycle in the basis contains exactly one edge which is not contained in any other cycle in the basis. In this paper we consider the problem of finding a Minimum Fundamental Cycle Basis (MINFCB) in a given graph, that is a FCB with minimum total cost.

Cycle bases have been used in the field of electrical networks since the time of Kirchoff [Kir47]. Fundamental cycle bases can be uniquely identified by their corresponding spanning trees, and can therefore be represented in a highly compact manner. The basic algebraic relations between spanning trees (chords and branches), fundamental cycles and fundamental cuts (the cuts identified by each of the tree branches) are presented in [SR61], p. 92-98. Besides the above-mentioned characterisation, Sysło established several structural results concerning FCBs [HS75, Sys78, Sys79, Sys81, Sys82a, Sys82b]. For example, two spanning trees whose symmetric difference is a collection of 2-paths (paths where each vertex, excluding the endpoints, has degree 2) give rise to the same FCB [Sys81]. Although the problem of finding a minimum cycle basis can be solved in polynomial time (see [Hor87] and the recent improvement by [AR03]), requiring fundamentality makes the problem $\mathcal{NP}$-hard [DPK82]. In [GA03], the problem of whether it is possible to find a polynomial-time approximation scheme (PTAS) for

[1]On leave from COPPE, Federal University of Rio de Janeiro, Brazil.

the MinFCB problem is addressed, and some approximation algorithms are discussed; a $4+\varepsilon$ approximation algorithm is presented for complete graphs, and a $2^{O(\sqrt{\log n \log\log n})}$ approximation algorithm for arbitrary graphs.

Interest in minimum FCBs arises in a variety of application fields, such as electrical circuit testing [BP01], periodic timetable planning [LM02], and generating minimal perfect hash functions [DKP95].

In this paper we propose three Integer Linear Programming (ILP) formulations for the MinFCB problem and we compare and discuss their relative advantages and disadvantages. For each formulation we consider its size and address the two following questions. For which size of instances does the formulation yield an optimal solution with a state-of-the-art commercial solver? How tight is the lower bound obtained by relaxing the integrality constraints of the formulation, and how fast are the corresponding linear programs solved? The computational results we have obtained with CPLEX 8.1 MIP solver show that even small MinFCB instances can be extremely challenging to solve to optimality. Since all but the smallest instances are out of reach of the formulation-oriented approach, we also propose a novel constructive heuristic, called the *C-order heuristic*, based on the idea of growing a spanning tree from the "center" of the graph, relegating most of the chords towards the "periphery" (intuitively, this gives rise to shorter cycles). Finally, we test this heuristic and compare it with an implementation of Deo's NT heuristic [DKP95] on several classes of graphs.

2 Formulations

In this section we present three different ILP formulations for the MinFCB problem. Recall that the graph $G=(V,E)$ is simple, undirected and biconnected. For all $i\in E$, let $w_i\geq 0$ be the cost of edge i. Let ν be the dimension of the cycle space of G, and $\delta(v)$ is the set of vertices adjacent to vertex v.

2.1 ILP Formulation 1

Consider the following binary variables:

$$\forall i\in E, k\leq\nu : x_{ik}=1 \quad \text{if edge } i\in \text{cycle } k \quad \text{and 0 otherwise} \tag{1}$$
$$\forall i\in E, k\leq\nu : t_{ik}=1 \quad \text{if edge } i \text{ is in cycle } k \text{ but not in spanning tree } T \quad \text{and 0 otherwise.} \tag{2}$$

The following is a valid ILP formulation of the MinFCB problem.

$$\begin{array}{llll}
\min & \sum_{i\in E}\sum_{k=1}^{\nu} w_i x_{ik} & & (3)\\
\text{s.t.} & \sum_{k=1}^{\nu} x_{ik}\geq 1 & \forall i\in E & (4)\\
& \sum_{i\in E} x_{ik}\geq 3 & \forall k\leq\nu & (5)\\
& \sum_{i\in E} t_{ik}=1 & \forall k\leq\nu & (6)\\
& \sum_{h=1}^{\nu} x_{ih}+\nu t_{ik}-2x_{ik}\leq \nu-1 & \forall i\in E, k\leq\nu & (7)\\
& t_{ik}\leq x_{ik} & \forall i\in E, k\leq\nu & (8)\\
& \sum_{i\in\delta(j)} x_{ik}\leq 2 & \forall j\in V, k\leq\nu & (9)\\
& \sum_{i\in\delta(j):i\neq h} x_{ik}\geq x_{hk} & \forall h\in\delta(j), j\in V, k\leq\nu & (10)\\
& x_{ik}\in\{0,1\} & \forall i\in E, k\leq\nu & (11)\\
& t_{ik}\in\{0,1\} & \forall i\in E, k\leq\nu. & (12)
\end{array}$$

The objective function minimizes the cost of all edges in the cycles of the FCB. By constraint (4), each edge belongs to at least one cycle; by constraint (5), each cycle consists of at least three edges (there are no parallel edges or loops in the graph). Constraint (6) ensures that each cycle has exactly one chord. By constraint (7), if an edge belongs to more than one cycle, then it cannot be considered a chord but it must be part of the spanning tree. Variables x and t are linked by constraint (8), which says that if an edge is a chord of cycle k then it must also belong to cycle k. Constraints (9) and (10) enforce a correct degree in the star of each vertex, with respect to each cycle incident on that vertex. This formulation has $2\nu m$ variables and $\nu O(m+n^2)$ constraints

2.2 ILP Formulation 2

In this formulation, we use flow-type constraints to identify a spanning tree rooted at vertex 1 (the choice of vertex 1 is arbitrary). The flow variables x identify a direction on each edge. We then impose conditions on the co-tree edges so that each cycle has exactly one chord. We also consider the edge costs W_{ij} for each edge $\{i,j\} \in E$ and take into account the edge stars $\gamma(v) = \{\{v,j\} \mid j \in \delta(v)\}$. Consider the following sets of binary variables:

$$\forall\{i,j\} \in E : x_{ij} = 1 \quad \text{if flow on edge } \{i,j\} \text{ is } i \to j \tag{13}$$

$$x_{ji} = 1 \quad \text{if flow on edge } \{i,j\} \text{ is } j \to i \quad \text{and 0 otherwise} \tag{14}$$

$$\forall\{i,j\} \in E : y_{ij} = 1 \quad \text{if edge } \{i,j\} \text{ is in } T \quad \text{and 0 otherwise} \tag{15}$$

$$\forall\{i,j\} \in E, k \le \nu : z_{ijk} = 1 \quad \text{if edge } \{i,j\} \text{ is in cycle } k \quad \text{and 0 otherwise} \tag{16}$$

$$\forall\{i,j\} \in E, k \le \nu : t_{ijk} = 1 \quad \text{if } y_{ij} = 1 \wedge z_{ijk} = 1 \quad \text{and 0 otherwise.} \tag{17}$$

Notice that variables t_{ijk} are meant to linearize the product $y_{ij}z_{ijk}$, and in fact mean "edge $\{i,j\}$ is the chord of cycle k. The following is a valid ILP formulation of the MINFCB problem.

$$\begin{array}{lrll}
\min & \sum_{\{i,j\}\in E} W_{ij} \sum_{k\le\nu} z_{ijk} & & (18)\\
\text{s.t.} & \sum_{k\le\nu} z_{ijk} \ge 1 & \forall\{i,j\} \in E & (19)\\
& t_{ijk} \le z_{ijk} & \forall\{i,j\} \in E, k \le \nu & (20)\\
& \sum_{\{i,l\}\in\gamma(j)} z_{ilk} \le 2 & \forall j \in V, k \le \nu & (21)\\
& \sum_{\{i,j\}\in E} z_{ijk} \ge 3 & \forall k \le \nu & (22)\\
& t_{ijk} \le y_{ij} & \forall\{i,j\} \in E, k \le \nu & (23)\\
& \sum_{\{i,j\}\in E}(z_{ijk} - t_{ijk}) = 1 & \forall k \le \nu & (24)\\
& \sum_{i\in\delta(j)} x_{ij} = 1 & \forall j \in V\backslash\{1\} & (25)\\
& \sum_{i\in\delta(0)} x_{i0} = 0 & & (26)\\
& y_{ij} = x_{ij} + x_{ji} & \forall\{i,j\} \in E & (27)\\
& \sum_{\{i,j\}\in E} y_{ij} = |V| - 1 & & (28)\\
& \sum_{k\le\nu}(z_{ijk} - t_{ijk}) \le 1 & \forall\{i,j\} \in E & (29)\\
& \sum_{\{i,l\}\in\gamma(j)} z_{ilk} \ge 2z_{hpk} & \forall j \in V, \{h,p\} \in \gamma(j), k \le \nu & (30)\\
& x_{ij} \in \{0,1\} & \forall(i,j) \in A = \{(i,j),(j,i)|\{i,j\} \in E\} & (31)\\
& y_{ij} \in \{0,1\} & \forall\{i,j\} \in E & (32)\\
& z_{ijk}, t_{ijk} \in \{0,1\} & \forall\{i,j\} \in E, k \le \nu & (33)
\end{array}$$

By constraint (19), all edges belong to at least one cycle; constraints (20) and (23) are linearization constraints. By constraint (21), at most two edges in each vertex star belong to any cycle; by constraint (22), we do not consider parallel edges or loops as cycles. Constraint (24) ensures that we have exactly one chord in each cycle. Constraints (25)-(28) make up for a tree-like structure where no flow enters node 1, but a flow of 1 enters each other node, and exactly $|V| - 1$ edges are considered for carrying the flow. By constraint (29), the same chord cannot appear in more than one cycle. Finally, constraint (30) says that if $\{h,j\}$ belongs to cycle k, then there must be some other edge, different from $\{h,j\}$, incident on vertex j.

This formulation has $(3 + 2\nu)m$ variables and $\nu O(m + n^2)$ constraints.

2.3 ILP Formulation 3

As in Section 2.2, we take into account the edge stars $\gamma(v) = \{\{v, j\} \mid j \in \delta(v)\}$ around each vertex v. We consider the following binary variables.

$$\forall i \in V, k \leq \nu : x_{ik} = 1 \quad \text{if vertex } i \text{ is in cycle } k \quad \text{and 0 otherwise} \tag{34}$$

$$\forall j \in E, k \leq \nu : y_{jk} = 1 \quad \text{if edge } j \in \text{cycle } k \quad \text{and 0 otherwise} \tag{35}$$

$$\forall j \in E, k \leq \nu : g_{jk} = 1 \quad \text{if edge } j \text{ is the chord of cycle } k \quad \text{and 0 otherwise.} \tag{36}$$

The following is a correct formulation for the MINFCB problem.

$$\begin{array}{lrll}
\min & \sum_{k\leq\nu}\sum_{j\in E} w_j y_{jk} & & (37)\\
\text{s.t.} & \sum_{j\in\gamma(i)} y_{jk} = 2x_{ik} & \forall i \in V, k \leq \nu & (38)\\
& \sum_{i\in V} x_{ik} \geq 1 & \forall k \leq \nu & (39)\\
& \sum_{k\leq\nu} y_{jk} \geq 1 & \forall j \in E & (40)\\
& \sum_{h\leq\nu, h\neq k} y_{jh} \leq y_{jk} + \nu g_{jk} - 1 & \forall j \in E, k \leq \nu & (41)\\
& \sum_{j\in E} g_{jk} \leq |E| - 1 & \forall k \leq \nu & (42)\\
& y_{jk} \geq 1 - g_{jk} & \forall j \in E, k \leq \nu & (43)\\
& x_{ik} \in \{0,1\} & \forall i \in V, k \leq \nu & (44)\\
& y_{jk} \in \{0,1\} & \forall j \in E, k \leq \nu & (45)\\
& g_{jk} \in \{0,1\} & \forall j \in E, k \leq \nu. & (46)
\end{array}$$

We leave the interpretation of the meaning of each constraint to the reader. This formulation contains $\nu(n + 2m)$ variables and $2\nu(2m + n + 1) + m$ constraints.

2.4 Computational Results: Solving MINFCB Formulations

For each of the above formulations, we are interested either in solving the MINFCB problem or in obtaining a lower bound to the optimal solution. We tackled each problem instance with the CPLEX 8.1 MIP solver. Likewise, we calculated lower bounds by solving a continuous relaxation of the ILP problem with the CPLEX 8.1 LP solver. All the tests have been carried out using a Pentium 4 2.66GHz with 1GB RAM running Linux.

We chose to test the above formulations on several instances of random euclidean graphs with edge costs equal to the distance between the two vertices adjacent to the edge (the vertices are positioned randomly in a 20×20 square centered at the origin; each edge has a predetermined probability p of being created). We have tried all instances of 5,6,7,8 vertices and 0.2, 0.4 and 0.6 edge creation probabilities. Larger instances were not taken into account because of the prohibitive sizes of the resulting ILP formulations.

From the results in Table 1 we can say that ILP Formulation 2 is very likely to be algebraically equivalent to ILP Formulation 1 (since the lower bound was exactly the same on the whole test suite), although the relaxation of ILP Formulation 1 is faster to solve. ILP Formulation 3 is not as tight as ILP Formulations 1 and 2; however, its relaxations are the fastest to solve.

The main conclusion which is immediately evident from the results presented in this section is that this approach to solving the MINFCB problem is doomed to failure for all but the smallest instances.

3 C-order Tree-growing Heuristic

As shown in Section 2.4, solving the MINFCB problem to optimality is a computationally challenging task. It therefore makes senses to look for heuristic algorithms. A common approach to designing heuristic algorithms for

	Formulations – Random euclidean weighted graphs											
	$p = 0.2$				$p = 0.4$				$p = 0.6$			
n	FCB cost	CPU	Bound	CPU	FCB cost	CPU	Bound	CPU	FCB cost	CPU	Bound	CPU
	ILP Formulation 1 (Section 2.1)											
5	34.9531	0.00	34.9531	0.00	125.24	0.09	113.142	0.00	96.1389	0.11	79.6077	0.00
6	61.0778	0.00	61.0778	0.00	102.322	0.05	88.3149	0.01	206.088	3022	149.934	0.01
7	63.1477	0.00	63.1477	0.00	-	-	237.994	0.02	-	-	123.321	0.02
8	127.921	1.69	104.496	0.00	-	-	246.490	0.04	-	-	267.256	0.08
	ILP Formulation 2 (Section 2.2)											
5	34.9531	0.00	34.9531	0.00	125.24	0.13	113.142	0.00	96.1389	0.2	79.6077	0.00
6	61.0778	0.00	61.0778	0.00	102.322	0.11	88.3149	0.00	-	-	149.934	0.01
7	63.1477	0.00	63.1477	0.00	-	-	237.994	0.04	157.517	10845	123.321	0.03
8	127.921	3.52	104.496	0.01	-	-	246.490	0.05	-		267.256	0.24
	ILP Formulation 3 (Section 2.3)											
5	34.9531	0.00	34.9531	0.00	125.24	0.05	93.2430	0.00	96.1389	0.04	67.5670	0.00
6	61.0778	0.00	61.0778	0.00	102.322	0.05	81.2510	0.02	-	-	122.256	0.01
7	63.1477	0.00	63.1477	0.01	-	0	165.936	0.01	-	-	96.2883	0.01
8	127.921	4.27	94.1546	0.00	-	0	180.825	0.01	-	-	178.901	0.03

Table 1: Computational results for formulation testing (CPU user time is expressed in seconds). Missing values are due to excessive ($>$ 4h) CPU time requirements.

the MINFCB problem is to construct a spanning tree with various FCB-minimizing criteria. Many tree-growing heuristics have been proposed in [DPK82, DKP95]. Tree-growing heuristics are generally very fast but may produce solutions with very sub-optimal FCB costs. In this section we shall introduce a novel tree-growing heuristic for the MINFCB problem based on a kind of vertex ordering that embeds global information about the whole graph. This ordering on the vertices is used to grow a a breadth-first spanning tree [Pat69]. The computational results (see Section 3.2) show that this heuristic produces solutions which on many instances have lower costs than those produced by existing heuristics. It performs particularly efficiently on uniform-cost mesh graphs (which have a lot of symmetries and are therefore very hard to tackle).

3.1 The C-order

Starting from the consideration that a spanning tree corresponding to the minimum FCB would include vertices having large star sizes, we shall now present a vertex ordering that aims to satisfy this condition as well as possible. To each vertex $u \in V$ we associate a sequence of integers which lists the star sizes of all the neighbours of u, then of all the neighbours of the neighbours of u, and so on at increasing distances from u. We then use these sequences to define an order on V. Here follows the formal construction of such an ordering.

For each $u \in V$ let $\bar{s}(u)$ be a sequence of $|V|$ pairs of numbers, each corresponding to a $v \in V$. More precisely, let the v-th component of the sequence be $\bar{s}_v(u) = \langle d(u,v), |\delta(v)| \rangle$ where $d(u,v)$ is the shortest distance between vertex u and vertex v and $|\delta(v)|$ is the cardinality of the edge star adjacent to vertex v. Order $\bar{s}(u)$ so that

$$\bar{s}_v(u) < \bar{s}_w(u) \Leftrightarrow d(u,v) < d(u,w) \vee (d(u,v) = d(u,w) \wedge |\delta(v)| > |\delta(w)|). \tag{47}$$

Let $s(u)$ be $\bar{s}(u)$ ordered according to rule (47). We shall now define an order $<_L$ on V (called the C-order) according to a natural order on the sequences $s(u)$. For all $i \leq n$ let $s_i(u)$ be the i-th pair in $s(u)$. Define $s_i(u) < s_j(u)$ according to the order on the pairs defined in (47). The corresponding order on the vertices becomes

$$u <_L v \Leftrightarrow \exists k \leq n (\forall l < k (s_l(u) = s_l(v)) \wedge s_k(u) < s_k(v)). \tag{48}$$

The ordering $<_L$ on V is such that the minimum element of V in that ordering has the property of "being in the center of graph" more than the other vertices. The C-ordering thus defines a kind of "baricenter" for any undirected graph G. By using this order we are able to discriminate among vertices "well inside the graph" and vertices "near the border". Intuitively, tree branches near the border of the graph force longer fundamental cycles.

The C-order is based on associating sequences of n components to each of the n vertices. The benefit of this choice is that the neighbour of each vertex is in fact the whole graph (thus global properties of the graph are taken into account). The disadvantage is that the C-ordering is sometimes computationally expensive to carry out (especially on graphs having many symmetries and unit edge weights).

3.2 Computational Results: the C-order Heuristic

We tested the tree-growing C-order heuristic on different classes of graphs.

Table 2 refers to several instances of random euclidean graphs with edge costs equal to the distance between the two vertices adjacent to the edge (the vertices are created randomly in a 20×20 square centered at the origin; each edge has a predetermined probability p of being created). We have tried all instances of 25,50,75,100 vertices and 0.3 and 0.7 edge creation probabilities. The results on Table 2 show that on random euclidean weighted graphs, the

	Tree-growing heuristics – Random euclidean weighted graphs							
	$p = 0.3$				$p = 0.7$			
n	FCB cost		CPU		FCB cost		CPU	
	C-Order	NT	C-Order	NT	C-Order	NT	C-Order	NT
25	4116.63	3400.41	0.004	0.009	6203.29	6283.11	0.006	0.007
50	17571.9	18943.2	0.014	0.013	27416.1	27372	0.025	0.027
75	28599.4	29353.8	0.033	0.027	70540.8	63864.5	0.096	0.121
100	58614	57749.8	0.082	0.014	115274	115807	0.559	0.953

Table 2: Computational results for C-order and Deo's NT tree-growing heuristics on random euclidean graphs. CPU timings are in seconds.

performances of C-order and Deo's NT [DKP95] heuristics are very similar, both in terms of quality of solution (for exactly half of the instances, the C-order heuristic found a better solution) and in terms of CPU time taken to find it.

The second class of test graphs we have considered is the square $n \times n$ mesh graph with unit costs on the edges. This is one of the most difficult testbeds for the MINFCB problem, both for heuristic and exact methods, due to the huge quantity of configurations having the same FCB cost. A unit cost square mesh graph with n vertices on each side has n^2 vertices and $2n(n-1)$ edges. Table 3 lists the FCB costs and CPU times of solutions found with C-order and with Deo's NT heuristics. On these graphs, the C-order heuristic performs better (albeit at a much higher CPU cost) than Deo's NT heuristic, in all instances.

For the sake of completeness, we have also tested the C-order tree-growing heuristic with the same graph test suite used in Table 1. The results are reported in Table 4.

4 Conclusion

In this paper we have presented three different ILP formulations for the MINFCB problem and we have compared their advantages and disadvantages by solving them and their continuous relaxations with CPLEX. We have then proposed a novel tree-growing heuristic for the MINFCB problem, based on a special vertex ordering (the C-order) that keeps track of the global properties of the graph, and compared its performance to another well-known constructive heuristic (Deo's NT heuristic). It turns out that the C-order heuristic outperforms the NT heuristic on unit cost mesh graphs in terms of FCB cost (but not of CPU time), whereas on random euclidean weighted graphs their performance is comparable, both in terms of FCB cost and CPU time.

	Tree-growing heuristics – Mesh graphs			
	C-order		Deo's NT	
n	FCB cost	CPU	FCB cost	CPU
5	72	0:00:00	78	0:00:00
10	492	0:00:00	518	0:00:00
15	1512	0:00:00	1588	0:00:00
20	3382	0:00:02	3636	0:00:00
25	6352	0:00:05	6452	0:00:00
30	10672	0:00:16	11638	0:00:00
35	16592	0:00:40	16776	0:00:00
40	24362	0:01:30	28100	0:00:01
45	34232	0:02:57	35744	0:00:01
50	46452	0:08:51	48254	0:00:03
55	61272	0:16:28	62026	0:00:04
60	78942	0:27:42	92978	0:00:06

Table 3: Computational results (FCB cost and CPU times expressed in hh:mm:ss)) on $n \times n$ mesh graphs having unit edge costs, with C-order and Deo's NT heuristics.

	C-order – Small random eucl. w. g.		
	$p = 0.2$	$p = 0.4$	$p = 0.6$
n	FCB cost	FCB cost	FCB cost
5	34.9531^*	125.24^*	101.27
6	61.0778^*	102.322^*	208.088
7	63.1477^*	217.965	157.517^*
8	127.921^*	402.184	366.073

Table 4: Computational results for C-order tree-growing heuristic on the same graph test suite used in formulation testing. CPU timings were between 0.001 and 0.004 seconds. Values marked with * are optimum values.

References

[AR03] E. Amaldi and R. Rizzi. Personal communication. 2003.

[BP01] A. Brambilla and A. Premoli. Rigorous event-driven (red) analysis of large-scale nonlinear rc circuits. *IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications*, 48(8):938–946, August 2001.

[DKP95] N. Deo, N. Kumar, and J. Parsons. Minimum-length fundamental-cycle set problem: New heuristics and an empirical investigation. *Congressus Numerantium*, 107:141–154, December 1995.

[DPK82] N. Deo, G.M. Prabhu, and M.S. Krishnamoorthy. Algorithms for generating fundamental cycles in a graph. *ACM Transactions on Mathematical Software*, 8(1):26–42, March 1982.

[GA03] G. Galbiati and E. Amaldi. On the approximability of the minimum fundamental cycle basis problem. *Workshop on Approximation and Online Algorithms (WAOA03), Budapest*, September 2003.

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

[HS75] E. Hubicka and M. Sysło. Minimal bases of cycles of a graph. In *Recent Advances in Graph Theory*, pages 283–293, Prague, 1975. 2nd Czech Symposium in Graph Theory, Academia.

[Kir47] G. Kirchhoff. über die auflösung der gleichungen, auf welche man bei der untersuchungen der linearen verteilung galvanisher ströme geführt wird. *Poggendorf Annalen Physik*, 72:497–508, 1847.

[LM02] Christian Liebchen and Rolf H. Möhring. A case study in periodic timetabling. In Dorothea Wagner, editor, *Electronic Notes in Theoretical Computer Science*, volume 66. Elsevier, 2002.

[Pat69] K. Paton. An algorithm for finding a fundamental set of cycles of a graph. *Communications of the ACM*, 12(9):514–518, 1969.

[SR61] S. Seshu and M.B. Reed. *Linear Graphs and Electrical Networks*. Addison-Wesley, Reading, MA, 1961.

[Sys78] M. Sysło. An efficient cycle vector space algorithm for listing all cycles of a planar graph. *Colloquia Mathematica Societatis János Bolyai*, pages 749–762, 1978.

[Sys79] M. Sysło. On cycle bases of a graph. *Networks*, 9:123–132, 1979.

[Sys81] M. Sysło. On some problems related to fundamental cycle sets of a graph. In R. Chartrand, editor, *Theory of Applications of Graphs*, pages 577–588, New York, 1981. Wiley.

[Sys82a] M. Sysło. On some problems related to fundamental cycle sets of a graph: Research notes. *Discrete Mathematics (Banach Centre Publications, Warsaw)*, 7:145–157, 1982.

[Sys82b] M. Sysło. On the fundamental cycle set graph. *IEEE Transactions on Circuits and Systems*, 29(3):136–138, 1982.