Operations Research Letters 10 (1991) 315–322
North-Holland August 1991

A polyhedral approach to edge coloring *

George L. Nemhauser and Sungsoo Park **
Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332–0205, USA

Received July 1990
Revised December 1990

We formulate the edge coloring problem on a simple graph as the integer program of covering edges by matchings. For the NP-hard case of 3-regular graphs we show that it is sufficient to solve the linear programming relaxation with the additional constraints that each odd circuit be covered by at least three matchings. We give an efficient separation algorithm for recognizing violated odd circuit constraints and a linear programming based constrained weighted matching algorithm for pricing. Computational experiments with the overall linear programming system are presented.

edge coloring, integer programming

1. Introduction

An *edge coloring* of a graph $G = (V, E)$ is a coloring of the edges of G so that any pair of edges that are incident to a common node have different colors. The *chromatic index* of G, denoted by $\chi(G)$, is the minimum number of colors in an edge coloring of G. The *edge coloring problem* is to find a coloring of G with $\chi(G)$ colors.

Let $\Delta(G)$ be the maximum degree over the nodes of G. Clearly, $\chi(G) \geq \Delta(G)$. The edge coloring problem for simple graphs, i.e. those without loops or parallel edges, appears to be greatly simplified by the following theorem of Vizing [15].

Theorem 1. *If G is a simple graph, then $\chi(G) = \Delta(G)$ or $\Delta(G) + 1$.*

In fact there is a proof of Vizing's theorem, see Faber, Ehrenfeucht and Kierstead [3] and Lovasz and Plummer [9] that gives a polynomial-time algorithm for coloring any simple graph with $\Delta(G) + 1$ colors. So the unresolved decision problem for a simple graph is to determine whether it can be colored with $\Delta(G)$ colors. As shown by Holyer [8], this problem is NP-complete, even for 3-regular graphs. Note that by Vizing's theorem, the decision problem for 3-regular graphs is just to determine whether the chromatic index is 3 or 4.

In this paper, we present a linear programming algorithm, involving both row and column generation, for solving the edge coloring problem. In Section 2, we formulate the problem as an integer program with a huge number of variables and consider its linear programming relaxation. In Section 3, we improve the formulation by providing additional linear inequalities that are satisfied by all integer solution but not all solutions to the original linear programming relaxation. In Section 4, we show that it suffices to add a simple, but large, class of these valid inequalities to solve the edge coloring problem for 3-regular graphs. Moreover we show that the separation problem for this class of inequalities is solvable in polynomial time for 3-regular graphs. Here the usual polynomial equivalence between separation and optimization does not hold because of the exponential number of variables. Section 5 presents a simplex algorithm that uses both row and column generation to solve a linear program with an exponential number of rows and columns. Computational results for 3-regular graphs are given in Section 6. The algorithm is very efficient when $\chi = 3$ but is considerably slower when $\chi = 4$.

* This work was supported in part by the National Science Foundation under Grant No. ECS-8719128.
** Present address: Department of Industrial Engineering, KAIST, Taejon, 305-701, Korea.

2. Fractional edge coloring

A *matching* on G is a subset of the edges with the property that no pair of edges in the subset is incident to a common node. Hence in an edge coloring, a set of edges can have the same color if and only if the set is a matching. Therefore the edge coloring problem is to find a minimum cardinality covering of the edges by maximal matchings, which can be formulated as the integer program

$$\text{(IP)} \quad \begin{aligned} &\chi(G) = \min 1x \\ &Ax \geq 1, \\ &x \geq 0 \text{ and integer,} \end{aligned}$$

where A is the edge-matching incidence matrix of G, i.e. $a_{ij} = 1$ if edge i is in the j-th matching and $a_{ij} = 0$ otherwise. In any optimal solution to (IP) $x_j = 0$ or 1, and the cover is defined to include the j-th matching if and only if $x_j = 1$.

As is the case with most combinatorial optimization problems, the edge coloring problem has many different formulations. For example, a formulation with a polynomial number of variables is obtained by letting $x_{jk} = 1$ if edge j receives color k and $x_{jk} = 0$ otherwise. A well-known difficulty in solving formulations of this type is the symmetry with respect to the colors.

We have two main reasons for studying the formulation (IP), although we do not know whether it is necessarily the best one for computational purposes. The first is our interest in the possibility of solving IP's using column generation. The second is because of the good approximation to $\chi(G)$ obtained from the linear programming relaxation of IP.

The *fractional edge coloring* problem, see Seymour [13] and Stahl [14], is obtained by dropping the integrality requirement in (IP), i.e.

$$\text{(LP)} \quad \begin{aligned} &\chi_{\mathrm{LP}}(G) = \min 1x \\ &Ax \geq 1, \\ &x \geq 0. \end{aligned}$$

Proposition 1. $\chi_{\mathrm{LP}}(G) \geq \Delta(G)$.

Proof. The result is a simple consequence of linear programming duality since the $\Delta(G)$ edges that are incident to some node of maximum degree yield a feasible solution to the dual. □

If $\chi_{\mathrm{LP}}(G) > \Delta(G)$, or if we have found an optimal solution to (LP) that is integral, then we also have found $\chi(G)$.

Proposition 2. *If* $\chi_{\mathrm{LP}}(G) > \Delta(G)$, *then* $\chi(G) = \Delta(G) + 1$, *and if* $\chi_{\mathrm{LP}}(G) = \Delta(G)$ *and there is an integral optimal solution to* (LP), *then* $\chi(G) = \Delta(G)$.

Proof. Both results follow immediately from $\chi_{\mathrm{LP}}(G) \geq \chi(G)$. □

Although (LP) contains a number of columns that grows exponentially with the size of G, it can be solved efficiently.

Theorem 2. *Problem* (LP) *can be solved in polynomial time by an ellipsoid algorithm.*

Proof. The dual of (LP), denoted by (DLP), is to find nonnegative edge weights with the property that the sum of the edge weights over each matching is not more than one and the sum over all edges is maximum. The feasibility of a nonnegative edge weight vector w to (DLP) can be checked by finding a maximum weight matching in G: if the maximum weight is not more than 1, then w is dual feasible; otherwise, a matching M that yields the maximum weight defines a dual constraint that is not satisfied by w.

Since maximum weight matching is solvable in polynomial time, see Edmonds [2], it follows by a theorem of Grötschel, Lovasz and Schrijver [6] relating the polynomial solvability of separation and linear programming that (DLP) and (LP) can be solved in polynomial time by an ellipsoid algorithm. □

Hence if the conditions of Proposition 2 hold, a minimum cardinality edge coloring can be found in polynomial time. The case that remains to be dealt with is when $\chi_{\mathrm{LP}}(G) = \Delta(G)$ but the only known optimal solutions to (LP) are fractional. Here our approach is to tighten (LP) by adding valid inequalities that are satisfied by all of the incidence vectors of edge colorings but not by the current fractional solution.

It has been demonstrated for a number of combinatorial optimization problems, see, for example, Hoffman and Padberg [7] and Nemhauser and Wolsey [10], that this approach can be very successful when the valid inequalities define facets, or at least high dimensional faces, of the convex hull of integral solutions. However, in all of the uses of constraint generation with which we are familiar, the number of columns is polynomially bounded. Thus what makes this formulation of the edge coloring problem interesting and novel from a mathematical programming perspective is the attempt to solve a linear program with an exponential number of columns (incidence vectors of matchings) and rows (facet-defining inequalities of the convex hull of edge colorings). Moreover, this formulation is a good candidate with which to explore the possibility of solving a 'doubly exponential' linear programming relaxation since it has a very small gap, i.e. $\chi(G) - \chi_{LP}(G) \leq 1$.

3. Valid inequalities for the edge coloring polyhedron

We call the convex hull of the set of feasible solutions to (IP) the *edge coloring polyhedron* and denote it by $P(G)$. Here we are interested in finding inequalities that are satisfied by all points in $P(G)$ but not all feasible solutions to (LP).

Let $U \subseteq V$ and $E(U)$ be the subset of edges with both ends in U. A matching can cover not more than $\lfloor \frac{1}{2} |U| \rfloor$ edges from $E(U)$. Hence, see Seymour [13] and Stahl [14], we get the family of valid inequalities for $P(G)$ given by

$$\sum_{\{j:\ M_j \cap E' \neq \emptyset\}} x_j \geq \left\lceil \frac{|E'|}{\lfloor \frac{1}{2} |U| \rfloor} \right\rceil$$

$$\text{for all } U \subseteq V \text{ and all } E' \subseteq E(U) \tag{1}$$

where x_j is the variable corresponding to the maximal matching M_j.

We are particularly interested in the case where $|U|$ is odd and the subgraph $C = (U, E')$ is a circuit. Then the right-hand side of (1) is $\lceil (2k + 1)/k \rceil = 3$ and we obtain the family of *odd circuit constraints* given by

$$\sum_{\{j:\ M_j \cap C \neq \emptyset\}} x_j \geq 3 \quad \text{for all odd circuits } C. \tag{2}$$

The odd circuit constraints are generally not implied by the edge constraints of (LP) unless $|C| = 3$, but they can be generated from linear combinations of the edge constraints and rounding. For some graphs they give facets of $P(G)$, see Park [12].

4. A separation procedure for 3-regular graphs

In this section we show that it is sufficient to add the odd circuit constraints to (LP) to solve the edge coloring problem for 3-regular graphs. This does no imply, however, that the odd circuit, edge and nonnegativity constraints define $P(G)$ when $P(G)$ is a 3-regular graph. However, when $\chi(G) = 4$, by adding odd circuit constraints we push the value of the optimal solution to (LP) above 3, which is sufficient.

Let (ALP) be the linear program obtained by augmenting (LP) with the odd circuit constraints (2) and let $\chi_{ALP}(G)$ be its optimal value.

Theorem 3. *If G is 3-regular and $\chi(G) = 4$, then $\chi_{ALP}(G) > 3$.*

Proof. We prove the contrapositive. Suppose $\chi_{ALP}(G) = 3$. Since $\chi(G) = 4$, every optimal solution to ALP is fractional. By hypothesis, in any such optimal solution x^*, we have $\Sigma x_j^* = 3$. Since $|E| = \frac{3}{2}|V|$, if $x_j^* > 0$ then the corresponding matching contains $\frac{1}{2}|V|$ edges, i.e. each column with positive weight corresponds to a perfect matching. Now delete one of the these perfect matchings with positive weight from G. What remains is a 2-factor, i.e. a subgraph G' that consists of a set of disjoint cycles. Moreover, the sum of the matching weights that cover the edges of G' is strictly less than 3.

There are 2 cases:

1. If all of the cycles of G' are even, then $\chi(G') = 2$ and the deleted matching gets a third color so that $\chi(G) = 3$, which is a contradiction.

2. If G' contains at least one odd cycle, then each of these cycles has the sum of the matching weights that cover it strictly less than 3. Thus the hypothesized optimal solution to (ALP) is not feasible because an odd circuit constraint is violated, which also is a contradiction.

Hence $\chi_{ALP}(G) > 3$. □

When G is 3-regular, $\chi_{LP}(G) = 3$ and the optimal solution to LP is fractional, we can use Theorem 3 to solve the separation problem for the odd circuit constraints. We delete a perfect matching with positive weight from G and then we either find a 3-coloring or a violated odd circuit constraint. Since the number of odd circuit constraints is bounded, this iterative procedure terminates finitely with a 3-coloring or a proof that 4-colors are required. In the later case a 4-coloring can be constructed efficiently as noted above. In the next section we give an algorithm that uses this separation scheme to solve (ALP) for 3-regular graphs.

Theorem 3 generalizes to graphs of any chromatic index. Let $\mathscr{G}_{k-1}$ be the family of graphs with maximum degree equal to $k-1$ and with chromatic index equal to k. For graph G the inequalities (2) generalize to

$$\sum_{\{j:\ M_j \cap H \neq \emptyset\}} x_j \geq k$$

$$\text{for all subgraphs } H \text{ of } G \text{ with } H \in \mathscr{G}_{k-1}. \qquad (3)$$

Now let (ALP_k) be the linear program obtained by augmenting (LP) with the constraints (3) and let $\chi_{\text{ALP}_k}(G)$ be its optimal value.

Theorem 4. *If $\Delta(G) = k$ and $\chi(G) = k+1$, then $\chi_{\text{ALP}_k} > k$.*

The proof is essentially the same as that of Theorem 3. However, whereas separation for $k = 3$ simply amounts to checking the components of a 2-regular graph for an odd circuit, separation for $k > 3$ appears to be much more difficult.

5. A row and column generation simplex algorithm to solve (ALP) for 3-regular graphs

To solve (ALP) by a simplex algorithm requires:

1. A separation routine for recognizing violated odd circuit constraints.

As discussed in Section 4, this routine is simple and fast since it just involves deleting a matching with positive weight and then checking whether the resulting subgraph of G, which is a collection of disjoint circuits, contains any odd circuits. (Note that all such odd circuits have at least 5 nodes since, as we remarked previously, the odd circuit constraints for triangles are implied by the constraints of (LP).) In fact, it is computationally efficient to repeat this procedure for each matching of positive weight until either a collection of all even disjoint circuits is found yielding a 3-coloring of G, or all such matchings have been considered. In the later case, all of the violated odd circuit constraints that have been identified can be added to the linear program. Our approach is to add them one at a time because it alleviates some of the difficulties associated with the column generation.

2. A pricing routine for finding a column to enter the basis or proving optimality.

When no odd circuit constraints are present the pricing problem is a weighted matching problem. However, when the linear program contains some odd circuit constraints, the pricing problem is a weighted matching problem with additional variables and constraints. In particular let w be the vector of dual variables on the edge constraints, u be the vector of dual variables on the existing odd circuit constraints. Then the pricing problem can be written as

$$\text{(PR)} \quad z(w, u) = \max \sum_{e \in E} w_e y_e + \sum_C u_C \pi_C$$
$$\sum_{e \in \delta(\{v\})} y_e \leq 1 \quad \text{for } v \in V,$$
$$\sum_{e \in E(S)} y_e \leq \tfrac{1}{2}(|S| - 1)$$
$$\text{for odd sets } S \subseteq V$$
$$\text{(blossom inequalities)},$$
$$\pi_C - \sum_{e \in C} y_e \leq 0 \quad \text{all existing circuits } C,$$
$$\pi_C \in \{0, 1\} \quad \text{all existing circuits } C,$$
$$y_e \in \{0, 1\} \quad \text{all } e \in E,$$

where $E(S) = \{e \in E\text{: both ends of } e \text{ in } S\}$.

The linear programming relaxation of (PR) with $0 \leq \pi_C \leq 1$ and $0 \leq y_e \leq 1$, denoted by (LPR), can be solved by a simplex algorithm using a separation routine for the blossom inequalities, see Padberg and Rao [11]. This approach has been shown to be computationally efficient for weighted matching problems by Grötschel and Holland [5]. A flowchart of the algorithm is given in Figure 1.

Let $\bar{z}(w, u)$ be the optimal value of (LPR). If $\bar{z}(w, u) \leq 1$, then the current solution is optimal to the given relaxation of (ALP) and therefore $\chi = 4$. On the other hand, if $\bar{z}(w, u) > 1$, we need either to find a feasible integer solution with weight > 1 to identify a column to enter the basis, or to establish that $z(w, u) \leq 1$ to prove optimality of the current solution. The work is already done for us if the solution found to (LPR) happens to be integral. If it is not integral, we first check for violated blossom inequalities and resolve the LP if any are found. If there are no violated blossom inequalities, we use the pivot and complement heuristic of Balas and Martin [1] to attempt to find an integral solution with weight > 1. If this fails we continue the solution of (PR) by branch-and-bound until we either find a feasible solution of weight > 1 or show that none exists.

6. Computational results

The algorithm is written in FORTRAN and was run on the IBM 9375 under VM/IS. Its major building blocks, XMP and ZOOM, consist, respectively, of a simplex method for linear programming and branch-and-bound routine for zero-one integer programming.

We have tested the algorithm on five sets of randomly generated 3-regular graphs, ranging from 20 to 60 nodes, and we report the results for five nontrivial graphs of each size. A graph G is said to

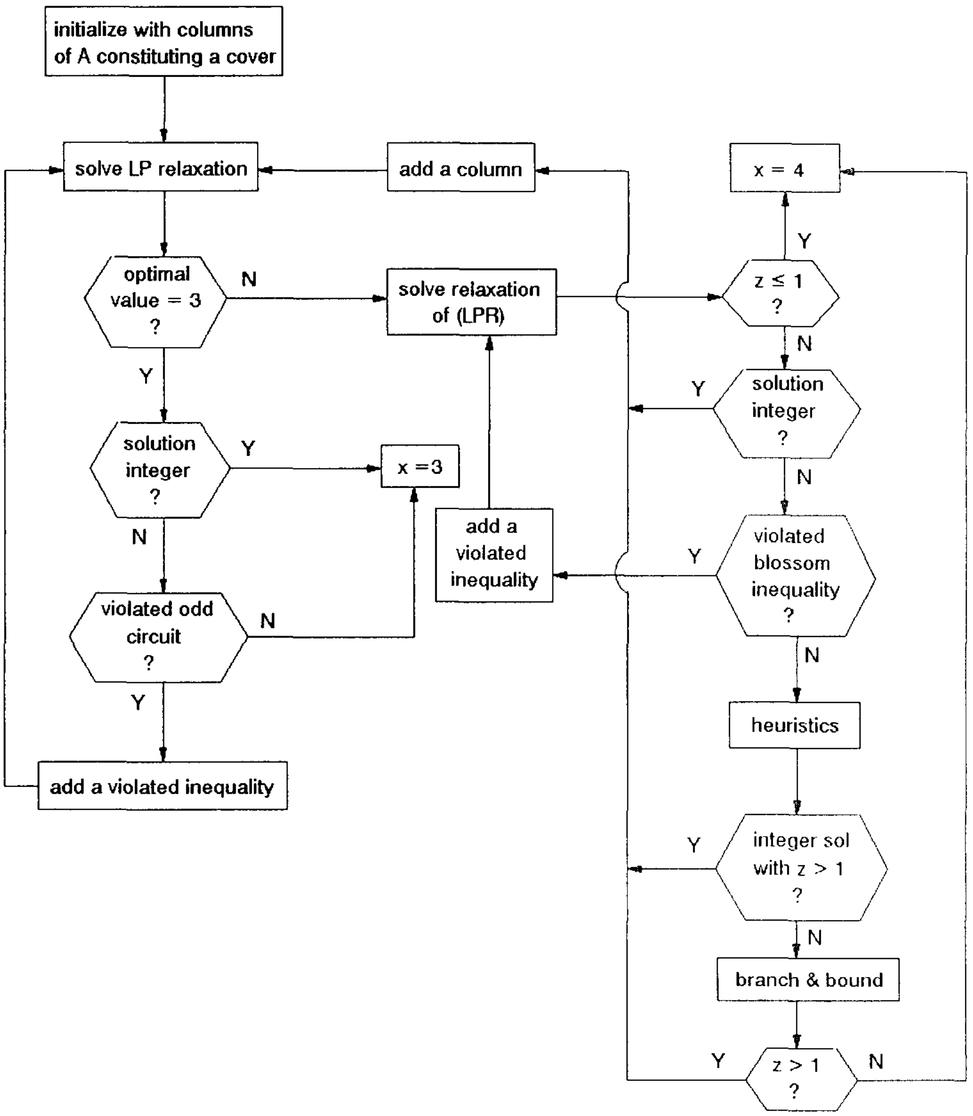

Fig. 1. Row and column generation simplex algorithm for 3-regular graphs.

be trivial if the solution to the initial linear program (without row or column generation) immediately yields $\chi(G) = 3$. In the random graphs each possible edge occurs with equal probability. In the process of generating such a graph, when a node reaches degree 3, the probability of choosing edges incident to it are set to zero.

We also have tested the algorithm on five 3-regular graphs whose chromatic indices are 4, namely the Petersen graph, the double-star snark and flower snarks with 12, 20 and 28 nodes. A snark is a 3-regular graph whose chromatic index is 4 and whose girth is at least five. For more details about these graphs, see Fiorini and Wilson [4].

Test results are shown in Table 1 and Table 2. Table 1 gives counts of cuts and columns generated. The number of generated columns is broken down between before and after odd circuit cut generation, and the after count is divided among those found in the LP, heuristic, and branch-and-bound phases. Table 2 gives the elapsed time for each part of the algorithm, i.e. solving the linear program, generating columns, and finding violated odd circuit constraints. For the random graphs, the first number in a cell is the maximum count or time, the second is the average and the third is the minimum over the five problems in the set.

For 3-colorable graphs, the computations can be stopped when a 2-factor consisting of even circuits has been identified, see column D of Table 2. But we have added all the violated odd circuit constraints, unless we can decide the chromatic index is 4, to see how many cuts are needed and if an integer solution can be found by adding the cuts. If the chromatic index is 4, we stop the computation without actually generating a coloring, although it would be trivial to do so.

Not surprisingly, the chromatic index for all of the randomly generated graphs turned out to be 3. Moreover, these problems are easy to solve since

Table 1
Compiled data for 3-regular graphs

Random graphs	# of odd circuit cuts	# of blossom cuts	Total # of columns generated	Before cuts	After cuts	LP phase	Heuristic phase	Branch-and-bound phase
1	0	6	10	10	0	0	0	0
$\lvert V \rvert = 20$	0	3.6	9.4	9.4	0	0	0	0
	0	2	8	8	0	0	0	0
2	15	13	40	19	23	7	9	7
$\lvert V \rvert = 30$	5.6	6	25.2	17.2	8	3.2	2.6	2.2
	1	2	18	15	1	0	0	0
3	9	14	34	24	13	8	4	4
$\lvert V \rvert = 40$	5.8	7.4	31.2	21.8	9.4	5.6	2.2	1.6
	3	3	28	20	4	0	0	0
4	25	51	81	31	51	16	26	11
$\lvert V \rvert = 50$	18.2	17	59.4	29	30.4	13.2	11.8	5.4
	10	5	43	27	14	10	1	1
5	22	31	74	33	41	24	13	5
$\lvert V \rvert = 60$	15.8	18.6	59	32.6	26.4	16.6	7.4	2.4
	7	8	44	32	12	11	1	0
Petersen $\lvert V \rvert = 10$	1	0	7	6	1	0	0	1
Double star $\lvert V \rvert = 30$	27	21	79	17	62	8	8	46
Flower snark $\lvert V \rvert = 12$	1	1	9	7	2	1	0	1
Flower snark $\lvert V \rvert = 20$	12	4	30	10	20	6	3	11
Flower snark $\lvert V \rvert = 28$	66	3	104	16	88	23	15	50

Table 2
Compiled times in seconds on the IBM 9375 for 3-regular graphs

Random graphs	Solving LP's (A)	Column generation (B)	Odd circuit cut finding (C)	Finding the chromatic index (D)	Total A + B + C
1	2.4	3.4	0	5.6	5.8
$\|V\| = 20$	2.3	3.2	0	5.2	5.5
	2.2	2.7	0	4.5	4.9
2	16.8	90.7	1.1	12.3	108.6
$\|V\| = 30$	7.8	33.4	0.6	11.1	41.8
	4.3	13.8	0.1	9.2	18.2
3	21.1	73.3	1.6	27.8	96.0
$\|V\| = 40$	15.9	45.6	0.8	22.0	62.3
	14.1	34.7	0.5	17.6	49.3
4	124.1	485.5	4.2	40.4	613.8
$\|V\| = 50$	79.8	207.7	3.3	36.6	290.8
	42.0	55.3	1.7	33.8	99.0
5	159.4	276.3	5.9	92.8	441.6
$\|V\| = 60$	108.9	181.6	4.6	62.0	295.1
	52.5	52.7	2.3	50.2	107.5
Petersen $\|V\| = 10$	2.1	2.7	≈ 0	4.8	4.8
Double star $\|V\| = 30$	52.1	574.4	1.4	627.9	627.9
Flower snark $\|V\| = 12$	2.1	3.6	≈ 0	5.7	5.7
Flower snark $\|V\| = 20$	7.0	51.3	0.5	58.8	58.8
Flower snark $\|V\| = 28$	288.6	788.6	4.5	1081.7	1081.7

in every instance the initial LP solution yielded a perfect matching with positive weight that was part of a 3-coloring. In other words, we can think of the LP solution as being an efficient and reliable indicator for determining some perfect matching in a 3-coloring, and once such a matching is determined no cuts are needed. On the other hand, the cuts were not very effective for determining integer solutions. We found integer solutions in solving the linear program for only three of the test problems, two in problem set 1 and one in problem set 2, which implies that the edge constraints and the odd circuit constraints are not close to describing $P(G)$ for the 3-colorable, 3-regular graphs.

Finally, we note that it generally takes more time to generate columns than to solve the linear program, and generally many cuts are needed to show that the snarks require 4 colors.

References

[1] E. Balas and R. Martin, "Pivot and complement: A heuristic for 0–1 programming", *Management Sci.* **26**, 86–89 (1980).

[2] J. Edmonds, "Maximum matching and a polyhedron with 0–1 vertices", *J. Res. Nat. Bur. Standards Sect. B* **69**, 125–130 (1965).

[3] V. Faber, A. Ehrenfeucht and H.A. Kierstead, "A new method of proving theorems on chromatic index", Los Alamos National Laboratories Preprint, LA-UR-82-661, 1981.

[4] S. Fiorini and R. Wilson, *Edge Colourings of Graphs*, Pitman, London, 1977.

[5] M. Grötschel and O. Holland, "Solving matching problems with linear programming", *Math. Programming* **33**, 243–259 (1985).

[6] M. Grötschel, L. Lovasz and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization", *Combinatorica* **1**, 169–197 (1981).

[7] K. Hoffman and M.W. Padberg, "LP-based combinatorial problem solving", *Ann. Operations Res.* **4**, 145–194 (1985).

[8] I. Holyer, "The NP-completeness of edge coloring", *SIAM J. Comput.* **10**, 718–770 (1981).

[9] L. Lovasz and M.D. Plummer, *Matching Theory*, Akademiai Kiado, Budapest, 1986.

[10] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York, 1988.

[11] M.W. Padberg and M.R. Rao, "Odd minimum cut sets and b-matchings", *Math. Oper. Res.* **7**, 67–80 (1982).

[12] S. Park, "Integer programming approach to the edge coloring problem", Ph.D. Dissertation, Cornell University, 1989.

[13] P. Seymour, "On multicolouring of cubic graphs and conjectures of Fulkerson and Tutte", *Proc. London Math. Soc.* (3) **38**, 423–460 (1979).

[14] S. Stahl, "Fractional edge coloring", *Cahiers Centre Études Rech. Opér.* **21**, 127–131 (1979).

[15] V.G. Vizing, "On an estimate of the chromatic class of a *p*-graph", *Diskret. Analiz.* **3**, 25–30 (1964).