

More on a Binary-Encoded Coloring Formulation

Jon Lee¹ and François Margot²

¹ IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, U.S.A.

jonlee@us.ibm.com

² Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA

fmargot@andrew.cmu.edu

Abstract. We further develop the 0/1 ILP formulation of Lee for edge coloring where colors are encoded in binary. With respect to that formulation, our main contributions are: (i) an efficient separation algorithm for general block inequalities, (ii) an efficient LP-based separation algorithm for stars (i.e., the all-different polytope), (iii) introduction of matching inequalities, and (iv) introduction of switched path inequalities and their efficient separation, (v) a complete description for paths, (vi) promising computational results.

Introduction

Let G be a simple finite graph with vertex set $V(G)$ and edge set $E(G)$ and let $m := |E(G)|$. For $v \in V(G)$, let $\delta(v) := \{e \in E(G) : e \text{ is incident to } v\}$. Let $\Delta(G) := \max\{|\delta(v)| : v \in V(G)\}$. Let c be a positive integer, and let $C := \{0, 1, \dots, c-1\}$.

A *proper edge c -coloring* of G is a function Φ from $E(G)$ to C , so that Φ restricted to $\delta(v)$ is an injection, for all $v \in V(G)$. Certainly, a proper edge c -coloring can not exist if $c < \Delta(G)$. Vizing [11] proved that a proper edge c -coloring always exists when $c > \Delta(G)$. Holyer [6] proved that it is NP-Complete to decide whether G has a proper edge $\Delta(G)$ -coloring (even when $\Delta(G) = 3$).

Lee [7] developed a 0/1 integer linear programming formulation of the feasibility problem of determining whether G has a proper edge c -coloring based on the following variables: For each edge $e \in E(G)$, we use a string of n 0/1-variables to encode the color of that edge (i.e., the n -string is interpreted as the binary encoding of an element of C). Henceforth, we make no distinction between a color (i.e., an element of C) and its binary representation in $\{0, 1\}^N$.

Let $N := \{0, \dots, n-1\}$. For $X \in \mathbb{R}^{E(G) \times N}$, we let x_e denote the row of X indexed by e and x_e^i denote the entry of X in row e and column i ($e \in E(G)$, $i \in N$). We define the *n -bit edge coloring polytope* of G as

$$Q_n(G) := \text{conv} \left\{ X \in \{0, 1\}^{E(G) \times N} : x_e \neq x_f, \forall \text{ distinct } e, f \in \delta(v), \forall v \in V(G) \right\}.$$

The graph G is a *star* if there is a $v \in V(G)$ such that $E(G) = \delta(v)$. If G is a star, then we call $Q(m, n) := Q_n(G)$ the *all-different polytope* (as defined in [7]). In this case, we let $M := E(G)$. For a general graph G , the all-different polytope

is the fundamental “local” modeling object for encoding the constraint that Φ restricted to $\delta(v)$ is an injection, for all $v \in V(G)$. Note that this type of constraint is present in several combinatorial problems besides edge coloring: vertex coloring, timetabling, and some scheduling problems for example. Although the focus of this paper is on edge coloring, the results of Sections 1 and 2 are relevant in all such situations.

For determining whether G has a proper edge c -coloring, we choose $n := \lceil \log_2 c \rceil$, so we are using only $\sim m \log c$ variables, while the more obvious assignment-based formulation would require mc variables. A rudimentary method for allowing only c of the possible 2^n colors encoded by n bits is given in [7]; A much more sophisticated method for addressing the case where the number of colors c is not a power of two (i.e., $2^{n-1} < c < 2^n$) can be found in [4].

One difficulty with this binary-encoded model is to effectively express the all-different constraint at each vertex — that is, to give a computationally-effective description of the all-different polytope by linear inequalities. In Sections 1 and 2, we describe progress in this direction. In Sections 3 and 4, we describe progress for general graphs (i.e., not just stars). In Section 5, we describe our implementation and results of computational experiments.

In the remainder of this section, we set some notation and make some basic definitions. For $x_e \in \mathbb{R}^N$ with $\mathbf{0} \leq x_e \leq \mathbf{1}$, and $S, S' \subseteq N$ with $S \cap S' = \emptyset$, we define the *value of (S, S') on e* as

$$\mathbf{x}_e(S, S') := \sum_{i \in S} x_e^i + \sum_{i \in S'} (1 - x_e^i).$$

When $S \cup S' = N$, the ordered pair (S, S') is a partition of N . A *t -light partition* for $x_e \in \mathbb{R}^N$ is a partition (S, S') with $\mathbf{x}_e(S, S') < t$. An *active partition* for $e \in E(G)$ is a 1-light partition. For $E' \subseteq E(G)$, we define the *value of (S, S') on E'* as

$$\mathbf{x}_{E'}(S, S') := \sum_{e \in E'} \mathbf{x}_e(S, S').$$

1 Separation for General Block Inequalities

For $1 \leq p \leq 2^n$, p can be written uniquely as

$$p = h + \sum_{k=0}^t \binom{n}{k}, \text{ with } 0 \leq h < \binom{n}{t+1}.$$

The number t (resp., h) is the *n -binomial size* (resp., *remainder*) of p . Then let

$$\kappa(p, n) := (t+1)h + \sum_{k=0}^t k \binom{n}{k}.$$

Let S, S' be a partition of N and let L be a subset of M . Then $X \in Q(m, n)$, must satisfy the *general block inequalities* (see [7]):

$$\text{(GBI)} \quad \kappa(|L|, n) \leq \mathbf{x}_L(S, S').$$

In fact, general block inequalities are facet-describing for the all-different polytope when the n -binomial remainder of $|L|$ is not zero [7].

Lemma 1. *Let p satisfy $1 \leq p \leq 2^n$, and let t be the n -binomial size of p . Then for $x_e \in \mathbb{R}^N$ with $\mathbf{0} \leq x_e \leq \mathbf{1}$, at most $4(n+1)^2 p^2 + (n+1)p$ partitions are $(t+1)$ -light for x_e .*

Proof. Let (S_1, S'_1) and (S_2, S'_2) be two $(t+1)$ -light partitions with $d := |S_1 \Delta S_2|$ maximum. Without loss of generality, we can assume that $S_1 = \emptyset$ by replacing x_e^i by $1 - x_e^i$ for all $i \in S_1$. As

$$2(t+1) > \mathbf{x}_e(S_1, S'_1) + \mathbf{x}_e(S_2, S'_2) \geq d,$$

we have that $d \leq 2t+1$. The number T of possible $(t+1)$ -light partitions satisfies

$$T \leq \sum_{k=0}^{2t+1} \binom{n}{k}.$$

Using that, for all $k \leq n/2$, $\binom{n}{2k} \leq \binom{n}{k}^2$ and $\binom{n}{2k-1} \leq \binom{n}{k}^2$ and that, for nonnegative numbers a, b , we have $a^2 + b^2 \leq (a+b)^2$, we get

$$T \leq \left(2 \sum_{k=0}^{t+1} \binom{n}{k} \right)^2 + \sum_{k=0}^{t+1} \binom{n}{k}$$

By hypothesis, we have $\binom{n}{t+1} \leq n \binom{n}{t} \leq np$, and thus

$$\sum_{k=0}^{t+1} \binom{n}{k} \leq (p-h) + np \leq (n+1)p.$$

The result follows. □

Note that computing all of the $(t+1)$ -light partitions for x_e in the situation of Lemma 1 can be done in time polynomial in p and n using Reverse Search [1]: The number of partitions in the output is polynomial in p and n , and Reverse Search requires a number of operations polynomial in p and n for each partition in the output.

We are led then to the following

SEPARATION ALGORITHM FOR GBI

- (0) Let $X \in [0, 1]^{M \times N}$, and let t be the n -binomial size of m .
- (1) For each $e \in M$, compute the set T_e of all $(t+1)$ -light partitions for x_e .
- (2) Then, for each partition (S, S') in $\cup_{e \in M} T_e$:
 - (2.a) Compute $F \subseteq M$ such that, for each $e \in F$, (S, S') is a $(t+1)$ -light partition for x_e .
 - (2.b) Order $F = \{e_1, \dots, e_f\}$ such that $\mathbf{x}_{e_i}(S, S') \leq \mathbf{x}_{e_{i+1}}(S, S')$ for $i = 1, \dots, f-1$.

- (2.c) If one of the partial sums $\sum_{i=1}^k \mathbf{x}_{e_i}(S, S')$, for $k = 2, \dots, f$ is smaller than $\kappa(k, n)$, then $L := \{e_1, \dots, e_k\}$ and (S, S') generate a violated GBI for X .

By Lemma 1, it is easy to see that the algorithm is polynomial in p and n . We note that a much simpler algorithm can be implemented with complexity polynomial in p and 2^n : Replace $\cup_{e \in E'} T_e$ by the set of all 2^n possible partitions.

Theorem 1. *Let $X \in [0, 1]^{M \times N}$. If the algorithm fails to return a violated GBI for X , then none exists.*

Proof. Suppose that $L \subseteq M$ generates a violated GBI for partition (S, S') . Let s be the n -binomial size of $|L|$. Observe that $\kappa(|L|, n) - \kappa(|L| - 1, n) \leq s + 1$. We may assume that no proper subset of L generates a GBI for partition (S, S') . Then $\mathbf{x}_e(S, S') < s + 1$ for all $e \in L$. As $s \leq t$, this implies that (S, S') is a $(t + 1)$ -light partition for x_e , and the algorithm will find a violated GBI. \square

We can sharpen Lemma 1 for the case of $t = 0$ to obtain the following result.

Lemma 2. *Let $e \in M$ and $x_e \in \mathbb{R}^N$ with $\mathbf{0} \leq x_e \leq \mathbf{1}$. Then there are at most two active partitions for e . Moreover, if two active partitions, say (S_1, S'_1) and (S_2, S'_2) exist, then $|S_1 \Delta S_2| = 1$.*

Proof. $2 > \mathbf{x}_e(S_1, S'_1) + \mathbf{x}_e(S_2, S'_2) \geq |S_1 \Delta S_2|$ implies that $|S_1 \Delta S_2| = 1$. Moreover, we can not have more than two subsets of N so that the symmetric difference of each pair contains just one element. \square

Motivated by Lemma 2, we devised the following heuristic as a simple alternative to the exact algorithm.

SEPARATION HEURISTIC FOR GBI

- (0) With respect to x_e , compute its (at most) two active partitions and their values.
- (1) Then, for each partition (S, S') of N :
 - (1.a) Compute the set T of elements in M that have (S, S') as an active partition.
 - (1.b) Sort the $e \in T$ according to nondecreasing $\mathbf{x}_e(S, S')$, yielding the ordering $T = \{e_1, \dots, e_t\}$.
 - (1.c) If one of the partial sums $\sum_{i=1}^k \mathbf{x}_{e_i}(S, S')$, for $k = 2, \dots, t$, is smaller than $\kappa(k, n)$, then $L := \{e_1, \dots, e_k\}$ and (S, S') generate a violated GBI for X .

The complexity is $O(2^n m \log m)$. Note that the heuristic is an exact algorithm if the n -binomial size of m is zero.

2 Exact LP-Based Separation

In this section we describe an exact separation algorithm for the all-different polytope $Q(m, n)$. The algorithm is polynomial in m and 2^n . In many situations (e.g., edge coloring), we consider 2^n to be polynomial in the problem parameters (e.g., $\Delta(G)$); so the algorithm that we describe may be considered to be efficient in such situations.

Theorem 2. *Let \bar{X} be a point in $[0, 1]^{M \times N}$. There is an efficient algorithm that checks whether \bar{X} is in $Q(m, n)$, and if not, determines a hyperplane separating \bar{X} from $Q(m, n)$.*

Proof. The point \bar{X} is not in $Q(m, n)$ if and only if there is a solution to:

$$(I.1) \quad \sum_{i \in M} \sum_{j \in N} \pi_i^j \bar{x}_i^j > \sigma ;$$

$$(I.2) \quad \sum_{i \in M} \sum_{j \in N} \pi_i^j x_i^j \leq \sigma , \quad \forall x \in Q_I(m, n),$$

where $Q_I(m, n) := Q(m, n) \cap \{0, 1\}^{M \times N}$, $\pi \in \mathbb{R}^{M \times N}$, and $\sigma \in \mathbb{R}$.

We apply the ellipsoid method (see [5]) to (I.1-2). This results in a polynomial-time algorithm provided that we can solve the separation problem for (I.2) in polynomial time. We can write this latter separation problem for $(\bar{\pi}, \bar{\sigma})$ as

$$(\mathcal{P}) \quad z := \max \left\{ \sum_{i \in M} \sum_{j \in N} \bar{\pi}_i^j x_i^j : X \in Q_I(m, n) \right\},$$

where $z > \bar{\sigma}$ if and only if there exists a violated inequality of the type (I.2).

But (\mathcal{P}) is equivalent to

$$(\mathcal{P}') \quad \max \sum_{i \in M} \sum_{k \in O} w_{ik} z_{ik}$$

$$\text{s.t.} \quad \sum_{i \in M} z_{ik} \leq 1 , \quad \forall k \in O ,$$

$$\sum_{k \in O} z_{ik} = 1 , \quad \forall i \in M ,$$

$$z_{ik} \in \{0, 1\}, \quad \forall i \in M, \quad \forall k \in O ,$$

where

$$O := \{0, 1, \dots, 2^n - 1\} ,$$

$$w_{ik} := \sum_{j \in N} \bar{\pi}_i^j \cdot \text{bit}_j[k] ,$$

$\text{bit}_j[k] :=$ the j th bit in the binary encoding of k .

The result follows since (\mathcal{P}') is just an ordinary assignment problem.

For computational purposes, we do not want to rely on the separation algorithm that is implied by the preceding proof, because it makes use of the ellipsoid method. We call a valid inequality $\sum_i \sum_j \pi_i^j x_i^j \leq \sigma$ *normalized* if $\sigma \leq 1$ and $-1 \leq \pi \leq 1$. Clearly, if a valid inequality separating \bar{X} from $Q(m, n)$ exists, then a normalized such inequality exists as well. In the complete version of this paper, we will describe a practical approach which is also theoretically efficient, using ideas similar to those of [9]. This approach yields an algorithm for producing maximally violated normalized cuts if any such cut exists. In Section 5, we refer to cuts produced in this way as *LP cuts* (LPC).

3 Matching Inequalities

Let S, S' be subsets of N with $S \cap S' = \emptyset$. The *optimal colors for* (S, S') are the colors $\mathbf{x} \in \{0, 1\}^N$ that yield $\mathbf{x}(S, S') = 0$. The set of optimal colors for (S, S') is denoted by $\mathcal{B}(S, S')$. Note that if (S, S') is a partition of N , then there is a unique optimal color which is the characteristic vector of S' . In general, if $|N \setminus (S \cup S')| = k$, then the set of optimal colors for (S, S') has 2^k elements (it is the set of vertices of a k -dimensional face of $[0, 1]^N$). Note that if $\mathbf{x} \in \{0, 1\}^N$ is not an optimal color for (S, S') , then $\mathbf{x}(S, S') \geq 1$.

Proposition 1. *Let $E' \subseteq E(G)$, and let $F \subseteq E'$ be a maximum matching in the graph induced by E' . Let (S, S') be a partition of N . The matching inequality (induced by E')*

$$(MI) \quad \mathbf{x}_{E'}(S, S') \geq |E' \setminus F|$$

is valid for $Q_n(G)$.

Proof. At most $|F|$ edges in E' can have the optimal color for (S, S') , and every other edge has a color contributing at least one to the left-hand side. \square

When E' is an odd cycle, the matching inequalities reduce to the so-called “type-I odd-cycle inequalities” (see [7] which introduced these latter inequalities and [8] which provided an efficient separation algorithm for them).

A MI is *dominated* if it is implied by MI on 2-connected non-bipartite subgraphs and by GBI. The following proposition shows that it is enough to generate the non-dominated MI, provided that the GBI generated by the separation heuristic for GBI of Section 1 are all satisfied.

Proposition 2. *Let G' be the graph induced by E' . The MI induced by E' is dominated in the following cases:*

- (i) G' is not connected;
- (ii) G' has a vertex v saturated by every maximum matching in G' ;
- (iii) G' has a cut vertex v ;
- (iv) G' is bipartite.

Proof. (i) The MI is implied by those induced by the components of G' .

(ii) The MI is implied by the MI on $G' - v$ and the GBI for $\delta(v) \cap E'$.

(iii) Let G_1 and G_2 be a partition of E' sharing only vertex v . By (ii), we can assume that there exists a maximum matching F of G with v not saturated by F . Then $E(F) \cap E(G_i)$ is a maximum matching in G_i for $i = 1, 2$. The MI is thus implied by the MI on G_1 and G_2 .

(iv) By the König's theorem, the cardinality of a minimum vertex cover of G' is equal to the cardinality k of a maximum matching F of G' . It is then possible to partition the edges of G' into k stars, such that star i has k_i edges. If the GBI inequalities for the stars are all satisfied, then summing them up yields:

$$\mathbf{x}_{\mathbf{E}(G')}(S, S') \geq \sum_{i=1}^k (k_i - 1) = |E(G')| - k = |E(G') \setminus M|,$$

and the MI induced by E' is also satisfied. \square

Recall that a *block* of a graph is a maximal 2-connected subgraph. Proposition 2 is the justification of the following:

SEPARATION HEURISTIC FOR MI

(0) Let \bar{X} be a point in $[0, 1]^{E(G) \times N}$.

(1) For each partition (S, S') of N :

(1.a) Compute the edges T for which (S, S') is an active partition.

(1.b) For each non-bipartite block of the graph G' induced by T :

(1.b.i) Compute a maximum matching $F(G')$ in G' .

(1.b.ii) Check if $\mathbf{x}_{\mathbf{E}(G')}(S, S') \geq |E(G') \setminus F(G')|$ is a violated matching inequality.

Complexity: Since each edge of G has at most two active partitions, all computations of active partitions take $O(nm)$ and all computations of non-bipartite blocks take $O(m)$. For one partition (S, S') , computing the maximum matchings takes $O(\sqrt{|V(G)|} m)$ [10]. The overall complexity is thus $O(2^n \sqrt{|V(G)|} m)$.

Note that ignoring edges e for which (S, S') is not an active partition does not prevent generation of violated matching inequalities: Suppose that e appears in a violated matching inequality $\bar{\mathbf{x}}_{\mathbf{E}(G')}(S, S') < |E(G') \setminus F(G')|$. Then $\bar{\mathbf{x}}_{\mathbf{E}(G')-e}(S, S') < |(E(G') - e) \setminus F(G' - e)|$ is also violated, as the left-hand side has been reduced by more than 1, while the right-hand side has been reduced by at most 1. The algorithm is nevertheless not exact, as we should generate MI for all 2-connected subgraphs, not only for blocks. In practice, the blocks are very sparse and rarely contain more than a few odd cycles. Enumerating the 2-connected non-bipartite subgraphs might thus be feasible.

4 Switched Walk Inequalities

Let $S, S' \subseteq N$ such that $S \cap S' = \emptyset$ and $|S \cup S'| \geq n - 1$. Then (S, S') is a *subpartition* of N .

Let (S_1, S'_1) be a subpartition of N . Let (S_2, S'_2) be a subpartition obtained from (S_1, S'_1) by performing the following two steps:

- (1) adding the only element not in $S_1 \cup S'_1$ (if any) either to S_1 or to S'_1 ; call the resulting partition (P_2, P'_2) .
- (2) removing at most one element from P_2 or at most one element from P'_2 .

Then (S_2, S'_2) is a *switch* of (S_1, S'_1) . Observe that $|\mathcal{B}(S_1, S'_1)| \leq 2$, that $|\mathcal{B}(S_2, S'_2)| \leq 2$ and that $|\mathcal{B}(S_1, S'_1) \cap \mathcal{B}(S_2, S'_2)| \geq 1$.

Let (e_1, \dots, e_k) be the ordered set of edges of a walk in G with $k \geq 2$. For $i = 1, \dots, k$, let (S_i, S'_i) be subpartitions of N such that

- (a) $|S_i \cup S'_i| = \begin{cases} n, & \text{if } i = 1, \text{ or } i = k; \\ n - 1, & \text{otherwise.} \end{cases}$
- (b) For $i = 1, \dots, k - 1$, (S_{i+1}, S'_{i+1}) is a switch of (S_i, S'_i) .
- (c) For all $j \in S_t$, if $[t+1, t']$ is a maximal interval such that for all $t+1 \leq i \leq t'$ we have $N - (S_i \cup S'_i) = \{j\}$, then $j \in S_{t'+1}$ if and only if $t' - t$ is odd.
- (d) For all $j \in S'_t$, if $[t+1, t']$ is a maximal interval such that for all $t+1 \leq i \leq t'$ we have $N - (S_i \cup S'_i) = \{j\}$, then $j \in S'_{t'+1}$ if and only if $t' - t$ is even.

Then the walk and the set of subpartitions $(S_1, S'_1), \dots, (S_k, S'_k)$ form a *switched walk*.

Given a switched walk, the inequality

$$(SWI) \quad \sum_{i=1}^k \mathbf{x}_{\mathbf{e}_i}(S_i, S'_i) \geq 1$$

is a *switched walk inequality*.

Example 1. Let $N := \{0, 1, 2\}$. Consider the path of edges $(e_1, e_2, e_3, e_4, e_5)$. Associated with the sequence of edges of the path is the switched walk: $(\{0\}, \{1, 2\})$, $(\{0\}, \{2\})$, $(\{1\}, \{2\})$, $(\{1\}, \{0\})$, $(\{1, 2\}, \{0\})$. The given switched walk gives rise to the SWI:

$$\begin{array}{rcccc} +x_1^0 & & +(1-x_1^1) & +(1-x_1^2) & \\ +x_2^0 & & & +(1-x_2^2) & \\ & & +x_3^1 & +(1-x_3^2) & \\ +(1-x_4^0) & & +x_4^1 & & \\ +(1-x_5^0) & & +x_5^1 & +x_5^2 & \geq 1 \end{array}$$

The only possibility for a 0/1 solution to violate this is to have $\mathbf{x}_{\mathbf{e}_i} = 0$ for $i = 1, 2, \dots, 5$. This implies that the color of e_1 must be 011. Then, of the two possible colors for e_2 (achieving $\mathbf{x}_{\mathbf{e}_2} = 0$), the only one that is different from the color of e_1 is 001. Similarly, e_3 must get color 101 and e_4 gets 100. But this is not different from the only color giving $\mathbf{x}_{\mathbf{e}_5} = 0$.

Theorem 3. *The SWI are valid for $Q_n(G)$.*

Proof. By induction on k . If $k = 2$, the SWI is a GBI and thus is valid.

Assume that there exists a SWI with $k \geq 3$ that is not valid for the edge coloring polytope. Then there exists a coloring of the edges in the walk W such that each edge $e_i \in W$ receives a color in $\mathcal{B}(S_i, S'_i)$. As subpartitions (S_i, S'_i) and (S_{i+1}, S'_{i+1}) are switches of each other, $|\mathcal{B}(S_i, S'_i) \cap \mathcal{B}(S_{i+1}, S'_{i+1})| \geq 1$, implying that once the color of e_i is chosen, then a unique color for e_{i+1} is available. As $\mathcal{B}(S_1, S'_1)$ contains a unique color, the coloring of edges in W is unique. It is then possible to transform each subpartition (S_i, S'_i) for $i = 2, \dots, k-1$ into a partition (T_i, T'_i) of N such that $\mathcal{B}(T_i, T'_i)$ is the color assigned to e_i .

We now prove by induction that $(T_{k-1}, T'_{k-1}) = (S_k, S'_k)$, yielding a contraction. If $k = 3$, then if j was the element missing in (S_2, S'_2) then conditions c) and d) above imply that j is in S_1 if and only if j is in S'_2 . Since j is in S_1 if and only if j is in T'_2 , we get $T'_2 = S'_2$ and we are done.

If $k > 3$, then the same reasoning for the switched walk starting with edge e_2 and partition (T_2, T'_2) and ending with e_k with partition (S_k, S'_k) yields the result by induction. \square

Next, we state a result indicating the importance of the switched walk inequalities. The proof will appear in the complete version of this paper.

Theorem 4. *If P is a path and $n \geq 2$, then $Q_n(P)$ is described by the SWI and the simple bound inequalities $\mathbf{0} \leq X \leq \mathbf{1}$.*

We separate the SWI by solving m shortest path problems on a directed graph with at most $8(n+1)m$ nodes and $32n(n+1)^2m$ edges. The overall complexity of the separation is $O(mn^3 \log(mn))$.

5 Computational Results

We report preliminary results for Branch-and-Cut (B&C) algorithms using the GBI, LPC, MI and SWI. The code is based on the open source BCP code with the open-source LP solver CLP [3] and was run on an IBM ThinkPad T-30. Test problems consist of

- (a) nine 4-regular graphs $g4_p$ on p nodes, for $p = 20, 30, \dots, 100$;
- (b) three 8-regular graphs $g8_p$ on p nodes, for $p = 20, 30, \dots, 40$;
- (c) the Petersen graph (*pete*);
- (d) two regular graphs on 14 and 18 vertices having overfull subgraphs (*of5_14_7* and *of7_18_9.5*);
- (e) a graph from [2] on 18 vertices and 33 edges (*jgt18*).

Graphs in (a) and (b) are randomly generated and can be colored with 4 or 8 colors respectively. It is likely that most heuristics would be able to color them optimally, but our B&C algorithms have no such heuristic, i.e. they will find a feasible solution only if the solution of the LP is integer. The remaining graphs are ‘‘Class 2’’ graphs, i.e. graphs G that can not be colored with $\Delta(G)$ colors.

A subgraph H of a graph G is an *overfull* subgraph if $|V(H)|$ is odd, $\Delta(H) = \Delta(G)$, and $|E(H)| > \Delta(H) \cdot (|V(H)| - 1)/2$. If G has an overfull subgraph, then G is a Class 2 graph. Graphs in (d) were randomly generated and have overfull subgraphs, but are not overfull themselves. The graph in (e) is a small non-regular Class 2 graph.

To illustrate the benefits and trade-offs between the different types of cuts, we report results of three B&C algorithms. The separation algorithms for the different types of cuts are: the separation heuristic for GBI of Section 1, the exact LPC separation algorithm alluded to at the end of Section 2, the heuristic separation for MI algorithm of Section 3 (except that blocks are not computed), and the separation algorithm for SWI of Section 4. The generation of the cuts is done as follows: GBI and LPC are generated as long as possible. MI, SWI and Gomory cuts are generated only during the first six rounds of cutting, but SWI are generated only if no GBI are found (at most one round of SWI are used at each node). Finally, LPC are generated only when no GBI, MI and SWI are found.

B&C 1 uses GBI, MI, and Gomory cuts. B&C 2 uses, in addition, LPC, and B&C 3 uses all five types of cuts. Table 1 gives the number of nodes in the enumeration tree. As expected, in general, the number of nodes is smaller when more cuts are in use, but for some problems, there is a big drop between variant 1 and 2, i.e. the use of LPC seems to be important. Most of these problems have relatively large degree, which is also expected, as GBI give a good approximation of the all different polytope when the number of edges is small. On the other hand, the use of SWI does not seem to help much on these problems.

Table 2 shows that for problem with low maximum degree, using SWI increases the overall cpu time. This (and Table 3) illustrates the difficulties for separating these inequalities efficiently. Even with the restricted use of one round of SWI cuts at most, the separation algorithm returns a large number of violated SWI cuts. A better understanding of these cuts might help generate “useful” ones more efficiently. The separation times are very small for GBI, MI, and Gomory cuts. The LPC, however take a significant time (about 30% of the total time for the 8-regular graphs and 50% for of7_18_9_5). The SWI separation is also time consuming, taking roughly 10% of the total time.

Table 1. Number of nodes.

	1	2	3
<i>g4_20</i>	13	17	21
<i>g4_30</i>	35	31	27
<i>g4_40</i>	41	45	35
<i>g4_50</i>	55	53	59
<i>g4_60</i>	61	63	67
<i>g4_70</i>	69	75	89
<i>g4_80</i>	75	83	81
<i>g4_90</i>	87	93	93
<i>g4_100</i>	107	125	105
<i>g8_20</i>	435	93	93
<i>g8_30</i>	7113	155	139
<i>g8_40</i>	5221	191	185
<i>pete</i>	5	7	7
<i>of5_14_7</i>	1537	955	879
<i>of7_18_9_5</i>	25531	8703	7377
<i>jgt18</i>	1517	1453	1263

Table 2. cpu time in seconds. A star denotes a problem not solved in 1hr.

	1	2	3
<i>g4_20</i>	0.50	1.00	1.60
<i>g4_30</i>	1.10	1.70	2.70
<i>g4_40</i>	1.90	3.50	5.30
<i>g4_50</i>	4.50	5.70	15.40
<i>g4_60</i>	5.90	8.70	28.20
<i>g4_70</i>	5.90	11.80	39.90
<i>g4_80</i>	7.60	14.90	43.70
<i>g4_90</i>	11.50	17.80	50.50
<i>g4_100</i>	12.10	30.10	82.60
<i>g8_20</i>	67.00	48.60	58.10
<i>g8_30</i>	*	130.10	151.60
<i>g8_40</i>	3366.20	244.90	320.60
<i>pete</i>	0.30	0.50	0.50
<i>of5_14_7</i>	103.30	107.20	111.40
<i>of7_18_9_5</i>	*	*	*
<i>jgt18</i>	107.50	126.20	139.70

Table 3. Number of generated cuts.

	1			2				3				
	GBI	MI	GOM	GBI	LPC	MI	GOM	GBI	LPC	MI	SWI	GOM
<i>g4_20</i>	72	2	22	124	0	6	31	102	0	0	631	37
<i>g4_30</i>	214	1	51	208	0	0	48	270	0	3	836	61
<i>g4_40</i>	282	3	70	328	0	1	70	316	0	4	1347	71
<i>g4_50</i>	524	13	102	598	0	7	96	850	0	15	4521	145
<i>g4_60</i>	770	4	132	688	0	8	128	1110	0	20	7058	173
<i>g4_70</i>	1038	10	141	971	0	16	151	1692	0	9	10169	232
<i>g4_80</i>	932	5	142	860	0	9	156	1780	0	16	9405	199
<i>g4_90</i>	1150	10	168	1054	74	7	168	1680	0	10	10560	206
<i>g4_100</i>	1414	4	183	2206	0	35	286	2260	0	8	16668	256
<i>g8_20</i>	10313	115	1248	2225	1349	23	216	2369	1460	20	3095	231
<i>g8_30</i>	338022	2633	24651	3934	2408	29	366	4036	2291	24	7236	352
<i>g8_40</i>	123843	2179	15527	4204	2820	31	441	5535	3043	39	11492	462
<i>pete</i>	0	8	19	0	0	8	19	0	0	8	0	19
<i>of5_14_7</i>	19838	680	4063	14690	3795	798	2995	14885	3305	777	8721	2899
<i>of7_18_9_5</i>	594021	12510	76323	286893	141606	7563	28801	254749	121398	7381	236250	25715
<i>jgt18</i>	13664	793	4305	13594	167	722	4138	12449	30	745	27714	3949

References

1. David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Appl. Math.*, 65(1-3):21–46, 1996. First International Colloquium on Graphs and Optimization (GOI), 1992 (Grimentz).
2. Amanda G. Chetwynd and Robin J. Wilson. The rise and fall of the critical graph conjecture. *J. Graph Theory*, 7(2):153–157, 1983.
3. COIN-OR. www.coin-or.org, 2003.
4. Don Coppersmith and Jon Lee. Parsimonious binary-encoding in integer programming. *IBM Research Report RC23838*, 2003.
5. Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1993.
6. Ian Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput.*, 10(4):718–720, 1981.
7. Jon Lee. All-different polytopes. *J. Comb. Optim.*, 6(3):335–352, 2002. New approaches for hard discrete optimization (Waterloo, ON, 2001).
8. Jon Lee, Janny Leung, and Sven de Vries. Separating type-I odd-cycle inequalities for a binary encoded edge-coloring formulation. *IBM Research Report RC22303*, 2002. to appear in: *Journal of Combinatorial Optimization*.
9. R. Kipp Martin. Using separation algorithms to generate mixed integer model reformulations. *Oper. Res. Lett.*, 10(3):119–128, 1991.
10. Vijay V. Vazirani. A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{VE})$ general graph maximum matching algorithm. *Combinatorica*, 14:71–109, 1994.
11. Vadim G. Vizing. On an estimate of the chromatic class of a p -graph. *Diskret. Analiz No.*, 3:25–30, 1964.