Top (2009) 17: 158–170
DOI 10.1007/s11750-009-0096-1

ORIGINAL PAPER

Applying mod-k-cuts for solving linear ordering problems

M. Oswald · G. Reinelt · H. Seitz

Received: 14 November 2008 / Accepted: 6 April 2009 / Published online: 9 May 2009

Abstract The linear ordering problem consists of finding an ordering of the nodes of the weighted complete digraph on n nodes such that the sum of the weights of the arcs compatible with the ordering is maximized. In this paper, we report about the usefulness of mod-k cuts in a branch-and-cut algorithm for solving linear ordering problems to optimality.

Keywords Linear ordering problem · Branch-and-cut · Mod-k cuts

Mathematics Subject Classification (2000) 90C27 · 90C57

1 Introduction

Let $D_n = (V_n, A_n)$ be a complete digraph on n nodes with $m = n(n-1)$ arcs and integer weights c_{ij} for every arc $(i, j) \in A_n$. A *tournament* T in A_n is a subset of arcs containing, for every pair of nodes i, j, either the arc (i, j) or the arc (j, i), but not both. An *acyclic* tournament is a tournament without directed cycles. The *weight of a tournament* is the sum of the weights of its arcs. It is easy to see that an acyclic tournament induces a linear ordering of the nodes and vice versa, and we can define the weight of a linear ordering as the weight of the associated tournament. The problem of finding an acyclic tournament (linear ordering) of maximal weight is called *linear ordering problem* (*LOP*). It is a classical NP-hard combinatorial optimization problem with several applications (Reinelt 1985). Another frequently used name is *triangulation problem* which refers to an equivalent formulation of the problem where

M. Oswald (✉) · G. Reinelt · H. Seitz
Institute of Computer Science, University of Heidelberg, Im Neuenheimer Feld 368,
69120 Heidelberg, Germany
e-mail: Marcus.Oswald@Informatik.Uni-Heidelberg.de

one looks for a simultaneous permutation of the rows and columns of a square matrix in order to maximize the sum of the superdiagonal elements.

In this paper, we deal with branch-and-cut algorithms for solving the *LOP* to optimality. Such algorithms focus on the *linear ordering polytope* P_{LO}^n which is defined as the convex hull of the characteristic vectors of the acyclic tournaments in A_n, i.e.,

$$P_{LO}^n = \text{conv}\{\chi^T \in \{0,1\}^m \mid T \subseteq A_n \text{ is an acyclic tournament}\},$$

where $\chi_{ij}^T = 1$, if $(i, j) \in T$, and $\chi_{ij}^T = 0$, otherwise. Obviously, there is a 1–1 correspondence between the vertices of P_{LO}^n and the acyclic tournaments of A_n.

Conceptually, the linear ordering problem can now be solved by maximizing $c^T x$ over P_{LO}^n (where c denotes the vector of arc weights). This is the basic idea of branch-and-cut algorithms. However, a description of P_{LO}^n with linear inequalities and equations is needed. Although this description exists, due to the NP-hardness of the problem, it is highly complex and one can only attempt to find part of it.

A coarse approximation of P_{LO}^n consists of the following equations and inequalities:

$$\begin{aligned} x_{ij} + x_{ji} &= 1, && \text{for all } i, j \in V_n, i < j, \\ x_{ij} + x_{jk} + x_{ki} &\leq 2, && \text{for all } i, j, k \in V_n, i < j, i < k, j \neq k, \\ x_{ij} &\geq 0, && \text{for all } i, j \in V_n. \end{aligned}$$

The equation system is the minimal equation system for P_{LO}^n, which implies that its dimension is $\binom{n}{2}$. The so-called *3*-dicycle inequalities define facets of the linear ordering polytope. The 0/1-points in this polytope are exactly the characteristic vectors of all acyclic tournaments. Therefore, by requiring in addition $x_{ij} \in \{0, 1\}$ for all $(i, j) \in A_n$, we obtain an integer programming formulation of the *LOP*. If instead of integrality conditions only the inequalities $x_{ij} \geq 0$ are introduced, this results in the so-called *3-dicycle relaxation*.

There has been a lot of work on studying the structure of the linear ordering polytope to improve the approximation by 3-dicycle inequalities (e.g., in (Doignon et al. 2006; Fiorini 2001; Grötschel et al. 1984; Leung and Lee 1994; Reinelt 1995)), and a first branch-and-cut algorithm has been described in Grötschel et al. (1985).

In this paper, we report on a branch-and-cut algorithm that does not make use of further a priori known inequalities but generates a type of inequalities of general nature. These inequalities are not specific for the *LOP*, and thus our approach can also be tried for other optimization problems. The algorithm starts with the 3-dicycle relaxation and then tries to strengthen it using the so-called *mod-k inequalities* which are generated based on the current fractional solution.

We assume familiarity of the reader with polyhedral combinatorics and the branch-and-cut approach. Section 2 discusses mod-k inequalities in general and the identification of maximally violated mod-k cuts. In Sect. 3, we describe the special case of mod-2 inequalities and their separation with shortest path algorithms. Section 4 discusses the usefulness of these general cuts based on experimental results for a set of benchmark problems.

2 Maximally violated mod-*k* cuts

As a first possibility to generate cuts in a branch-and-cut algorithm we describe the so-called mod-k inequalities which are a special kind of *Chvátal–Gomory cuts* (Caprara et al. 2000).

Let $Ax \leq b$ be a system of linear inequalities with integral coefficients and let $k > 1$ be an integer number. Suppose, we scale every inequality p of $Ax \leq b$ by a nonnegative factor μ_p and sum up the resulting inequalities.

Now, if μ denotes the vector of all factors μ_p, assume that all coefficients of $\mu^T A$ are divisible by k and that the remainder of dividing $\mu^T b$ by k is $k-1$. Then μ satisfies the congruence system

$$\mu^T A \equiv 0 \bmod k,$$
$$\mu^T b \equiv k-1 \bmod k.$$

We have $\mu^T b = sk + (k-1)$ for some $s \in \mathbb{Z}$, and therefore $\mu^T b - (k-1)$ is divisible by k. Furthermore, $\mu^T Ax$ is divisible by k for all integer vectors x, and therefore the inequality $\mu^T Ax \leq \mu^T b - (k-1)$ is valid for all feasible integer solutions of $Ax \leq b$.

We can express the inequality in an equivalent way as *mod-k inequality*

$$\frac{1}{k}\mu^T Ax \leq \frac{1}{k}\big(\mu^T b - (k-1)\big).$$

Now consider some fractional solution x^* of $Ax \leq b$. Recall that all coefficients of this inequality are integral. In a branch-and-cut algorithm we would like to find an integer k and a vector μ such that the congruence system above is satisfied and the corresponding mod-k inequality is violated by x^*, thus providing a cutting plane (a so-called *mod-k cut*). Since $\mu^T Ax^* \leq \mu^T b$, this solution can violate $\frac{1}{k}\mu^T Ax \leq \frac{1}{k}(\mu^T b - (k-1))$ by at most $\frac{k-1}{k}$ and the maximal violation can only be achieved if $\mu^T Ax^* = \mu^T b$, i.e., if $\mu_p = 0$ for all p with $A_{p\cdot}x^* < b_p$, where $A_{p\cdot}$ denotes the pth row of A.

We describe an algorithm for the separation of maximally violated mod-k cuts as suggested in Caprara et al. (2000). Let x^* be a fractional solution of the current LP relaxation $Ax \leq b$. Due to the remarks above, in order to find a maximally violated inequality, we restrict the congruence system to contain only those inequalities that are tight for x^*. We have to choose k and find an integer multiplier vector $\mu \geq 0$ such that the congruence system is satisfied. Note that the coefficients of μ can obviously be restricted to values smaller than k, i.e., $\mu_p \in \{0, 1, \ldots, k-1\}$ for all p. Furthermore, due to a theorem of Caprara et al. (2000), only prime numbers have to considered for k.

If k is a prime number, then the solution of the congruence system can be obtained by applying standard Gaussian elimination to a modified system over the finite field $\mathbb{F}_k = \mathbb{Z}/k\mathbb{Z}$ with k elements $\{0, 1, \ldots, k-1\}$. To this end, we transform the integral coefficients of A and b into elements of $\mathbb{F}_k$ by defining $\overline{z} = z \mod k = z - \lfloor z/k \rfloor k$, for $z \in \mathbb{Z}$.

Now let $\overline{A} = (\overline{a}_{ij})$ and $\overline{b} = (\overline{b}_{ij})$. Every solution vector μ for the resulting system

$$\mu^T \overline{A} = 0^T \quad \text{and} \quad \mu^T \overline{b} = k-1 \tag{1}$$

over $\mathbb{F}_k$ also solves the system of congruences.

Suppose the system (1) has no solution. In this case, we might be satisfied with a mod-k cut with less than maximal violation and could try to find a solution of the modified system

$$\mu^T \overline{A} = 0^T \quad \text{and} \quad \mu^T \overline{b} = q, \quad \text{for } q \in \{1, \ldots, k-2\}.$$

But as every solution μ over $\mathbb{F}_k$ yields a solution $((k-1)/q)\mu$ of (1) over $\mathbb{F}_k$, it holds that if we have a mod-k cut violated by l/k, $l \leq k-2$, then there exists a maximally violated mod-k cut.

In a more common way, (1) can be written as

$$\overline{A}^T \mu = 0 \quad \text{and} \quad \overline{b}^T \mu = k-1. \tag{2}$$

The rows of $\overline{A}^T$ correspond to arcs, whereas the columns represent valid inequalities that are tight for x^*. Let us define $\tilde{A}^T = \left(\frac{\overline{b}^T}{\overline{A}^T}\right)$, i.e., $\tilde{A}^T$ is $\overline{A}^T$ with $\overline{b}^T$ as 0th row. After having transformed $\tilde{A}^T$ into an upper triangular form by Gaussian elimination, one can easily check whether there exists a maximally violated mod-k cut or not. Usually the existence of one cut gives rise to plenty maximally violated mod-k cuts. The number f of free variables μ_i, these are variables the value of which can be chosen freely from $\{0, 1, \ldots, k-1\}$, ranges from dozens to a few hundreds. There exist exactly k^f different solutions of (2) and, even though the mapping of solutions to cuts is not injective, we will have to address the problem of selecting cuts from the huge set of generated constraints to be added to the linear relaxation.

3 Mod-2 cuts

We next describe the efficient exact separation procedure for the case $k = 2$ given in Caprara and Fischetti (1996). Again, we search for a multiplier vector μ for $Ax \leq b$ satisfying the congruence system (2). The arguments of Sect. 2 show that

$$\frac{1}{2}\mu^T Ax \leq \frac{1}{2}\left(\mu^T b - 1\right) \tag{3}$$

is valid for $\{x \in \mathbb{Z}^n \mid Ax \leq b\}$ and can be violated by at most $\frac{1}{2}$ by a tight fractional solution x^* of $Ax \leq b$. For $\lambda = \frac{1}{2}\mu$ and μ satisfying the congruence system, all entries of $\lambda^T A$ are integers and $\lambda^T b - \frac{1}{2} = \lfloor \lambda^T b \rfloor$. Therefore, (3) can be expressed equivalently as

$$\lambda^T Ax \leq \left\lfloor \lambda^T b \right\rfloor.$$

This inequality is the so-called $\{0, \frac{1}{2}\}$-*Chvátal–Gomory inequality*. We now describe an algorithm searching for a violated $\{0, \frac{1}{2}\}$-Chvátal–Gomory cut for a given fractional solution of $Ax \leq b$ which is not restricted to binding inequalities as in the

previous section. The identification of a suitable vector λ, in principle, amounts to optimizing over

$$P_{1/2} = \left\{ x \in \mathbb{R}^n \mid Ax \leq b, \lambda^T Ax \leq \lfloor \lambda^T b \rfloor \text{ and } \lambda \in \left\{0, \frac{1}{2}\right\}^m \text{ such that } \lambda^T A \in \mathbb{Z}^n \right\}.$$

The separation problem for $P_{1/2}$ is NP-hard. There are, however, possibilities to relax $P_{1/2}$ in such a way that optimization in polynomial time is possible.

The system $Ax \leq b$ is transformed into a weakened system $A'x \leq b'$ containing at most two odd coefficients per row. To this end, let $O_i = \{j \mid a_{ij} \text{ is odd}\}$ be the index set of all odd coefficients in row i. Furthermore, let $-x_{ij} \leq 0$ be the lower bound and $x_{ij} \leq d_j$ be the upper bound of the variables.

The *L-weakening* makes use of the lower bound constraints $x_{ij} \geq 0$. For $h, g \in O_i$, $h < g$, the corresponding L-weakening of an inequality $\sum_j a_{ij} x_{ij} \leq b_i$ is

$$a_{ih} x_{ih} + a_{ig} x_{ig} + \sum_{j \notin O_i} a_{ij} x_{ij} + \sum_{j \in O_i \setminus \{h,g\}} (a_{ij} - 1) x_{ij} \leq b_i.$$

Using upper bounds $x_{ij} \leq 1$ in an analogous way, we obtain for $h, g \in O_i$, $h < g$, the *U-weakening*

$$a_{ih} x_{ih} + a_{ig} x_{ig} + \sum_{j \notin O_i} a_{ij} x_{ij} + \sum_{j \in O_i \setminus \{h,g\}} (a_{ij} + 1) x_{ij} \leq b_i + \sum_{j \in O_i \setminus \{h,g\}} d_j.$$

Both weakenings are applied to all inequalities in $Ax \leq b$ with $|O_i| \geq 3$, hence the transformed system $A'x \leq b'$ contains $\binom{|O_i|}{2}$ L-weakenings and $\binom{|O_i|}{2}$ U-weakenings for every inequality in $Ax \leq b$. Generally, it even has exponentially many rows, but, due to Caprara and Fischetti (1996), the system $A'x \leq b'$ can be reduced because, for every triple (i, h, g), it is sufficient to consider only two weakenings with respect to a fractional solution x^* of $Ax \leq b$. Namely, only the weakenings with minimum slack and even (odd right-hand side, respectively) have to be taken into account. They can be computed in linear time $O(n)$.

Hence, we have found a way to optimize over the relaxation

$$P'_{1/2} = \left\{ x \in \mathbb{R}^n \mid A'x \leq b', \lambda A'x \leq \lfloor \lambda^T b' \rfloor \text{ and } \lambda \in \left\{0, \frac{1}{2}\right\}^m \text{ such that } \lambda^T A' \in \mathbb{Z}^n \right\}$$

of $P_{1/2}$ in polynomial time.

Mod-2 cuts for the linear ordering problem can be identified in the following way. Let $Ax \leq b$ be the system of all 3-dicycle inequalities and x^* a solution of this system. We first construct from $Ax \leq b$ a new inequality system $A'x \leq b'$ where each inequality has a left-hand side with at most two odd coefficients. This is achieved by applying the above procedure. For example, the 3-dicycle inequality $x_{ij} + x_{jk} - x_{ik} \leq 1$ is replaced by the U-weakenings

$$\begin{aligned} 2x_{ij} + x_{jk} - x_{ik} &\leq 2, \\ x_{ij} + 2x_{jk} - x_{ik} &\leq 2, \\ x_{ij} + x_{jk} - 2x_{ik} &\leq 1, \end{aligned}$$

and the L-weakenings

$$x_{jk} - x_{ik} \leq 1,$$
$$x_{ij} - x_{ik} \leq 1,$$
$$x_{ij} + x_{jk} \leq 3.$$

In the following, let I_p denote the pth inequality of $A'x \leq b'$ and a'_{pij} the coefficient of x_{ij} in I_p and let

$$E_{\text{odd}} = \{(ij, kl) \mid \exists\, I_p \text{ with odd right-hand side such that } a'_{pij} \text{ and } a'_{pkl} \text{ are odd}\},$$
$$E_{\text{even}} = \{(ij, kl) \mid \exists\, I_p \text{ with even right-hand side such that } a'_{pij} \text{ and } a'_{pkl} \text{ are odd}\}.$$

We construct the weighted graph $G = (V, E)$ with

$$V = \{ij \mid 1 \leq i, j, \leq n, i < j\},$$
$$E = E_{\text{odd}} \cup E_{\text{even}}.$$

The edge weights of G are the slacks of $A'x \leq b'$ with respect to x^*, i.e., the edge weight of (ij, kl) is the slack of its corresponding inequality I_p.

We now have to find the shortest cycle in G with an odd number of edges in E_{odd}. To this end, we construct the weighted graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ with nodes

$$\widetilde{V} = \{ij \mid ij \in V\} \cup \{\widetilde{ij} \mid ij \in V\}$$

and edges $\widetilde{E} = E_1 \cup E_2$, where

$$E_1 = \{(\widetilde{ij}, kl), (ij, \widetilde{kl}) \mid (ij, kl) \in E_{\text{odd}}\},$$
$$E_2 = \{(\widetilde{ij}, \widetilde{kl}), (ij, kl) \mid (ij, kl) \in E_{\text{even}}\}.$$

The weights of $\widetilde{G}$ are adopted from the edges of G. In $\widetilde{G}$, we now search for the shortest path between an arbitrary node ij and its counterpart $\widetilde{ij}$.

Let $P = (ij, \ldots, \widetilde{ij})$ be such a shortest path with length strictly less than 1. We add up all inequalities whose corresponding edges are contained in P. Let this new inequality be $d^T x \leq \delta$. Since P contains an odd number of edges in E_1, the right-hand side δ is odd. Furthermore, as every node in P represents an odd coefficient in A' and as all inequalities corresponding to edges in P are summed up, every odd coefficient is added twice, hence all coefficients of the left-hand side of the new inequality are even. Dividing $d^T x \leq \delta$ by 2 and rounding down the right-hand side, we obtain the inequality

$$\frac{1}{2} d^T x \leq \left\lfloor \frac{\delta}{2} \right\rfloor$$

with integer coefficients which is valid for P^n_{LO}. If the length of P is strictly less than 1, then this inequality is violated by x^*. It is maximally violated by the amount of $\frac{1}{2}$ if x^* satisfies all inequalities in the combination with equality.

The advantage of the shortest path mod-2 method is that there is no restriction on the constraints that are used for the generation of the cut, as long as they can be transformed in a way that they have exactly two odd coefficients on the left-hand side. Another advantage is that the graph $\widetilde{G}$ is sparse, which is convenient for the shortest path computation. On the other hand, no extension from 2 to bigger prime numbers is possible, and the violation of the resulting cut is maximal if and only if all used constraints are tight for x^*.

It was shown in Caprara and Fischetti (1996) that the family of shortest path mod-2 cuts with respect to 3-dicycle inequalities contains a certain subclass of Möbius ladder inequalities, which constitute a rich constraint class of facets of the linear ordering polytope. In fact, besides for 3-dicycle facets, this is the only known separation algorithm or heuristic for a class of facet-defining inequalities of P_{LO}^n. There is a rich knowledge about the facet structure of this polytope, but the development of further separation procedures is still a research task. Only for some classes it was shown that the separation problem is NP-hard.

4 Computational experiments

We have applied our branch-and-cut algorithm on ten random problem instances `p40-01` – `p40-10` ($n = 40$) and on `p50-01` ($n = 50$). Details on these problems and further benchmarks can be found in LOLIB (2008). All computations were performed on a PC equipped with 2 Xeon E5450 processors, 2.5 GHz, 6 MB Cache and 8 GB RAM.

Since we want to assess the usefulness of mod-k cuts and mod-2 cuts, we first computed the upper bounds obtained from the 3-dicycle relaxation. This relaxation was solved with a cutting plane algorithm using the branch-and-cut framework Abacus (Jünger and Thienel 2000) and the LP solver Cplex 8.1 (Ilog Cplex 2009).

It should be noted that the solution of the 3-dicycle relaxation can be speeded up considerably if violated 3-dicycle cuts are added as follows: We first generate all violated cuts and then sort them with respect to their angle with the objective function. (In the 3-dicycle case, it is sufficient to just consider the sum of the three objective function coefficients.) Then the cuts with the largest sum, i.e., with the smallest angle, are added to the linear program. The best number depends on the size of the problem, for the instances considered here we added at most 250 3-dicycle inequalities per iteration.

The problem instances are difficult and no 3-dicycle relaxation had an integral optimum solution. In the following, we describe how we deal with the fractional solution x^* of the relaxation. We will also speak about the digraph associated with x^* which consists of the arcs whose associated variables have a positive value.

4.1 Rotation of facets

Since the generation of cuts is time consuming, we incorporated a heuristic element in our separation procedures to possibly generate further cuts cheap. Our idea is based on an interesting property of P_{LO}^n.

Let $P = \text{conv}(X) \subseteq \mathbb{R}^d$ be a polytope. An affine mapping ψ of $\mathbb{R}^d$ onto itself is called a *rotation mapping* of P if $X = \psi(X)$. It is clear that a rotation mapping transforms a facet F of P to a facet $\psi(F)$ of P.

For the linear ordering polytope there are two rotation mappings. The *arc reversal mapping* ϕ (Reinelt 1985) is defined by

$$\phi(x)_{ij} = x_{ji}, \quad \text{for all } 1 \leq i, j \leq n.$$

This mapping transforms a facet $f^T x \leq f_0$ to a facet $g^T x \leq g_0$, where $g_{ij} = f_{ji}$, for all i and j, and $g_0 = f_0$. In Bolotashvili et al. (1999), a second mapping ψ^r is presented. For an arbitrary fixed $r \in \{1, \ldots, n\}$, it is defined by

$$\begin{aligned}
\psi^r(x)_{rj} &= x_{jr}, \quad \text{for all } 1 \leq j \leq n,\ j \neq r,\\
\psi^r(x)_{jr} &= x_{rj}, \quad \text{for all } 1 \leq j \leq n,\ j \neq r,\\
\psi^r(x)_{ij} &= x_{ij} + x_{jr} + x_{ri} - 1, \quad \text{for all } 1 \leq i, j \leq n,\ i \neq r,\ j \neq r.
\end{aligned}$$

It is shown in Bolotashvili et al. (1999) that if $f^T x \leq f_0$ defines a facet F of P^n_{LO}, then

$$\begin{aligned}
\sum_{i=1, i\neq r}^{n} \sum_{j=1, j\neq r}^{n} f_{ij}\psi^r(x_{ij}) &= \sum_{i=1, i\neq r}^{n} \sum_{j=1, j\neq r}^{n} \left(f_{ij}(x_{ij} + x_{jr} - x_{ir}) + f_{ir}x_{ri} + f_{ri}x_{ir}\right)\\
&\leq f_0
\end{aligned}$$

defines the facet $\psi^r(F)$.

We enhanced our separation routines by using rotation in the following way. For a given LP-solution x^* and a fixed rotation parameter $r \in \{1, \ldots, n\}$, we compute the mapping $\psi^r(x^*)$. Now we perform all separation procedures directly on $\psi^r(x^*)$. If a violated inequality is found, then we construct its rotated version which is then violated by the original LP solution x^*. This procedure is performed for every r in a random order.

Note that using the rotation mappings also for the 3-dicycle separation does not lead to additional inequalities because 3-dicycles are mapped to other 3-dicycles or trivial inequalities.

A mapping of the vertices of P^n_{LO} onto themselves was already presented in McLennan (1990). But, since this mapping is only well-defined for permutations, it cannot be applied in our case, where the fractional LP solution x^* has to be transformed.

4.2 Generation of mod-k cuts

We use two general strategies for trying to generate mod-k cuts violated by the current fractional solution x^*. The first strategy considers small subdigraphs (of the digraph defined by x^*) and generates all violated cuts that can be found for this digraph. The idea is that by proceeding this way, the relaxation can locally be strengthened considerably and therefore allow for a reasonable bound improvement in the branch-and-cut

algorithm. The second strategy applies the mod-k separation routine to the complete digraph and selects cuts afterwards. Cuts for the whole digraph should provide global information which is also important for the algorithm. We implemented two strategies of the first and one of the latter type.

The strategies for choosing an appropriate subdigraph differ as follows. The *variable heuristic* limits the number n_V of variables from which the subdigraph is constructed, while the *improvement heuristic* limits the number n_N of the nodes of the subdigraph.

The variable heuristic starts with the variable that occurs in most constraints of the current LP. (If there are several variables satisfying this condition, then we choose one of them at random.) Then we continue to successively select all other variables from these constraints and also choose all constraints that contain these new variables. If the limit n_V on the number of variables is reached, the selection is stopped and the subdigraph G' is defined by the variables selected by this procedure.

A practical problem we had to address was that already a slight increase of n_V could lead from subdigraphs for which no cuts were found to subdigraphs where very many violated cuts existed. This phenomenon turned out to be caused by the fact that not all 3-dicycle inequalities for the subdigraph are part of the LP relaxation. After all trivial and binding 3-dicycle inequalities were added, this problem disappeared. We experienced that for our problem instances $n_V = \frac{1}{4}n$ and $n_V = \frac{1}{5}n$ are good bounds as the resulting digraph was just big enough to generate a reasonable number of mod-k cuts.

The improvement heuristic uses information from the current LP solution. For every node, the sum of the values of variables corresponding to outgoing arcs is computed. Then the nodes are linearly ordered with respect to nonincreasing sums. Our heuristic is based on the hypothesis that nodes closely together in this ordering will also be neighboring in an optimum solution. For forming the subdigraph we therefore take the first n_N nodes of the ordering, then the second n_N nodes, etc., and generate all mod-k cuts from these n/n_N subgraphs. Values $n_N \in \{\frac{1}{4}n, \frac{1}{5}n, \frac{1}{6}n\}$ lead to suitable results.

The third strategy is to apply the maximally violated mod-k method to the whole digraph and select cuts afterwards. This selection strategy is based on some different criteria. First, we randomly order the columns of the matrix to avoid to always generate the first cuts obtained from the system (2) in each iteration. Second, we prefer cuts introducing few non-zero coefficients because dense LPs are usually more difficult than sparse ones. A further criterion is to select only the single basic solution or at most f trivial solutions of (2) and not take all k^f possible solutions of the system into account.

A detailed presentation of further aspects of cut generation and selection for mod-k separation is given in Fricke (2007).

4.3 Generation of mod-2 cuts

For the shortest path calculation of the mod-2 procedure, we use Dijkstra's algorithm leading to running time $O(\widetilde{E} \log \widetilde{V})$. To avoid that inequalities are found more than once, we delete all nodes that are part of an already found violated inequality from the list of potential starting nodes.

In the case of rotation, we repeat this procedure for all rotation parameters r in a random order until the limit of 250 cuts is reached. For the violation tolerance, we use the relatively high value of 0.01 because otherwise the separation procedure finds a lot of cuts without significantly improving the dual bound. In addition, we use tailing off which stops the mod-2 separation if the last 10 separations did not improve the bound by more than 0.05 percent.

4.4 Computational results

The aim of this paper is to study how much mod-k separation can improve the branch-and-cut algorithms which are only based on the 3-dicycle relaxation. For measuring this improvement, we compute the *gap closure*

$$100 \cdot \frac{|c^T x^* - c_{3\text{cycle}}|}{|c_{\text{opt}} - c_{3\text{cycle}}|},$$

where c_{opt}, $c_{3\text{cycle}}$, and $c^T x^*$ are the optimum objective function value, the 3-dicycle upper bound, and the bound obtained with additional mod-k cuts, respectively. Thus the gap closure gives (in percent) how much of the gap between 3-dicycle bound and optimum value could be closed.

Table 1 displays for the instances `p40-01` – `p40-10` the optimum values, the 3-dicycle bound and the improved bound when additional cuts are added for tightening the LP relaxation. In the first two experiments, we added mod-2 and mod-k cuts separately (Mk and M2), in the third experiment both separations were employed (Mk + M2). The branch-and-cut algorithm was stopped when no more violated inequalities could be found at the root node, i.e., no branching was started.

Table 2 verifies that these bounds can be improved considerably without much additional computational effort when rotation is employed to find further cuts.

Improvement with respect to the 3-dicycle relaxation is easier to assess if one considers the gap closure. Table 3 shows that the gap between 3-dicycle bound and optimum can be closed by 86% on average and that the gap closure is mainly due to mod-2 cuts with rotation.

Table 1 Root bounds

Problem	Opt	3-cyc	Mk	M2	Mk + M2
`p40-01`	9540	9577.48	9571.09	9540.00	9540.00
`p40-02`	7767	8317.00	8252.75	8053.25	8053.11
`p40-03`	8822	9115.33	9058.53	8859.77	8859.59
`p40-04`	8064	8622.67	8544.46	8300.85	8298.93
`p40-05`	5808	6571.33	6449.60	6213.72	6207.33
`p40-06`	10787	11255.30	11176.90	10944.40	10939.30
`p40-07`	8971	9598.00	9484.08	9243.53	9239.43
`p40-08`	8388	8991.67	8865.77	8584.46	8581.07
`p40-09`	8097	8702.00	8597.85	8417.49	8410.87
`p40-10`	9177	9587.67	9529.26	9337.68	9334.69

Table 2 Root bounds with rotation

Problem	Opt	3-cyc	Mk_R	M2$_R$	Mk + M2$_R$
p40-01	9540	9577.48	9568.69	9540.00	9540.00
p40-02	7767	8317.00	8241.53	7933.32	7932.84
p40-03	8822	9115.33	9051.45	8822.00	8822.00
p40-04	8064	8622.67	8517.68	8153.76	8153.77
p40-05	5808	6571.33	6443.01	6037.71	6036.77
p40-06	10787	11255.30	11169.30	10809.20	10809.10
p40-07	8971	9598.00	9471.06	9073.02	9072.75
p40-08	8388	8991.67	8856.02	8432.82	8433.41
p40-09	8097	8702.00	8596.59	8216.73	8217.30
p40-10	9177	9587.67	9530.65	9218.55	9218.37

Table 3 Gap closure

Name	Mk	Mk_R	M2	M2$_R$	Mk + M2	Mk + M2$_R$
p40-01	17.05%	23.45%	100.00%	100.00%	100.00%	100.00%
p40-02	11.68%	13.72%	47.96%	69.76%	47.98%	69.85%
p40-03	19.36%	21.18%	87.12%	100.00%	87.19%	100.00%
p40-04	14.00%	18.79%	57.61%	83.95%	57.95%	83.94%
p40-05	15.95%	16.81%	46.85%	69.99%	47.69%	70.03%
p40-06	16.74%	18.36%	66.39%	95.26%	67.48%	95.28%
p40-07	18.17%	20.25%	56.53%	83.73%	57.19%	83.77%
p40-08	20.86%	22.47%	67.46%	92.58%	68.02%	92.48%
p40-09	17.21%	17.42%	47.03%	80.21%	48.23%	80.01%
p40-10	14.22%	13.88%	60.87%	89.88%	61.60%	89.93%

So, with respect to bound improvement, the additional separation has proved its advantages. But interesting as well is to check if this improvement also leads to faster computation times when a provably optimum solution has to be computed.

Our experiments revealed that the separation of maximally violated mod-k inequalities did not have a big effect on the root bound and is, in general, not worth the effort compared to mod-2 separation. Mod-2 separation gives a better gap closure, and because of the better bounds the number of branch-and-nodes needed to solve the problems to optimality is considerably smaller (even when rotation is not invoked). Therefore, we did not use mod-k separation anymore. Table 4 displays the respective results of our computations.

Table 4 shows the following facts. If only 3-dicycle separation is used, the algorithm spends most of its CPU time for solving the linear programs. Mod-2 separation changes this relation. Now separation is responsible for the CPU time. Because of the better bounds, the number of branch-and-cut nodes is drastically decreased. However, since separation is time consuming, the overall solution time for the problems with

Table 4 CPU times (sec), numbers of subproblems and percentages of separation time

Name	3-cyc			M2			M2$_R$		
	time	#subs	sep	time	#subs	sep	time	#subs	sep
p40-01	0	3	30%	1	1	70%	1	1	75%
p40-02	417	5441	4%	1586	147	87%	2091	41	91%
p40-03	8	115	6%	32	5	79%	23	1	79%
p40-04	198	2317	4%	768	69	86%	836	19	89%
p40-05	6096	24317	3%	5379	471	85%	6792	73	93%
p40-06	48	609	4%	182	19	84%	123	3	84%
p40-07	262	2775	3%	947	85	87%	725	15	87%
p40-08	117	1287	4%	381	33	85%	285	3	87%
p40-09	462	5625	4%	1917	177	87%	1181	23	89%
p40-10	33	421	4%	194	19	86%	221	7	87%
p50-01	2838	8349	7%	4542	153	82%	1474	47	76%

$n = 40$ is not reduced. But this changes for larger problems. Problem p50-01 can now be solved in half of the time.

Our computational experiments for the linear ordering problem lead us to the conclusion that the incorporation of general cut generation procedures is worthwhile and promising, and should also be tried for other combinatorial optimization problems. This coincides with the suggestion in Andrello et al. (2007), where a heuristic mod-2 separation procedure was applied to different ILPs. The optimal use of mod-2 and mod-k inequalities still has to be explored. To some extent, their potential cannot be fully exploited because LPs are becoming more difficult, at least for current LP solvers.

References

Andrello A, Caprara A, Fischetti M (2007) Embedding $\{0, 1/2\}$-cuts in a branch and cut framework: a computational study. Inf J Comput 19(2):229–238

Bolotashvili G, Kovalev M, Girlich E (1999) New facets of the linear ordering polytope. SIAM J Discrete Math 12:326–336

Caprara A, Fischetti M (1996) $\{0, \frac{1}{2}\}$-Chvátal–Gomory cuts. Math Program 74:221–235

Caprara A, Fischetti M, Letchford AN (2000) On the separation of maximally violated mod-k cuts. Math Program 87:37–56

Doignon J-P, Fiorini S, Joret G (2006) Facets of the linear ordering polytope: a unification for the fence family through weighted graphs. J Math Psychol 50:251–262

Fiorini S (2001) Determining the automorphism group of the linear ordering polytope. Discrete Appl Math 112:121–128

Fricke L (2007) Maximally violated mod-k cuts: a general purpose separation routine and its application to the linear ordering problem. Diploma thesis, University of Heidelberg

Grötschel M, Jünger M, Reinelt G (1984) A cutting plane algorithm for the linear ordering problem. Oper Res 32:1195–1220

Grötschel M, Jünger M, Reinelt G (1985) Facets of the linear ordering polytope. Math Program 33:43–60

Ilog Cplex (2009) http://www.ilog.com/products/cplex

Jünger M, Thienel S (2000) The ABACUS system for branch-and-cut-and-price algorithms in integer programming and combinatorial optimization. Softw Pract Exp 30:1325–1352. http://www.informatik.uni-koeln.de/abacus

Leung J, Lee J (1994) More facets from fences for linear ordering and acyclic subgraph polytopes. Discrete Appl Math 50:185–200

LOLIB (2008) A library of linear ordering benchmark instances. http://www.informatik.uni-heidelberg.de/groups/comopt/software/LOLIB

McLennan A (1990) Binary stochastic choice. In: Chipman J, McFadden D, Richter M (eds) Preferences, uncertainty and rationality. Westview Press, Boulder, pp 187–202

Reinelt G (1985) The linear ordering problem: algorithms and applications. Research and exposition in mathematics, vol 8. Heldermann

Reinelt G (1995) A note on small linear ordering polytopes. Discrete Comput Geom 10:67–78