Improving the LP bound of a MILP by dual concurrent branching and the relationship to cut generation methods

H. Georg Büsching
Händelstraße 9
32457 Porta Westfalica
Germany
lpbuesching@googlemail.com

April 9, 2011

Abstract

In this paper branching for attacking MILP is investigated. Under certain circumstances branches can be done concurrently. By introducing a new calculus it is shown there are restrictions for certain dual values and reduced costs. As a second unexpected result of this study a new class of cuts for MILP is found, which are defined by those values. This class is a superclass of all other classes of cuts. Furthermore the restrictions of the dual values and the reduced costs can be used for studying the addition of arbitraries inequalities.

1 Motivation of the following thoughts

Nowadays the technique for doing MILP (Mixed Integer Linear Programming) is based on the branch and bound method. This method uses the best solution of the linear inequality system with objective function (= LP-instance) by leaving out the integer conditions from the mixed integer linear inequality system with objective function (= MILP-instance). Then this method searches for an 0-1 (or integer) variable x_n, which has a non-integer value q_n. The next step is to create two new LP-instances by adding first $x_n = 0$ (or $x_n \leq [q_n]$) and secondly $x_n = 1$ ($x_n \geq [q_n + 1]$). By continuing this process a binary tree of problems is created.
Now take two different nodes in this tree, so you look at two different LP-instances. With both problems it is possible that some x_n is still not integer. We'll create a branch on that variable for both problems. It can happen at a big and sparse MILP-instance, that the same similar branching will lead to exactly the same calculations at the new LP-instances. From a numerical point of view

this is unsatisfactory.

We present here some new ideas which use some kind of independence of branching. These will help to prevent such double calculations. One further aim of these new techniques is a better measurement and control of what happens at a branching. A practical and short-term outcome should be better limits for huge MILP-instances. The prominent group of huge Traveling Salesman Problems is a part of this group. This group was indeed the starting point of the author's thoughts about this topic.
We'll show that the combination of branches can be described by an ordinary linear inequality system, so that the problem to get an optimal combination of branches will be a LP-instance (luckily not a MILP-instance). We'll reach this formulation at the middle of the second section at theorem 2.7. Instead of a binary tree of depth n, which has 2^n problems, we want to use just $2n$ problems. We'll try to combine the solutions of the $2n$ problems as well as possible to get a bound for the original problem (MILP-instance), which will be better than the LP-bound but normally not as good as the bound by solving all 2^n problems.
We'll furthermore see that it is even possible to define a very huge LP for each MILP, which represents the ability to combine the several case differentiations.

The main idea:

We'll measure the differences of the dual variables and the gain of the objective function when creating new problems, which each has one inequality more than the starting LP-instance. These differences of the dual variables are naturally connected to the branches. Then we'll choose those differences of dual variables, so that for all combinations of choices at the connected branches, all dual inequalities will hold. By adding the gain of each chosen branching, we get a total gain, which gives a better limit of the original problem.

The article is mostly self-contained, the only real reference are the basics about LPs as in [1]. If you are only interested in the theory of the defined cuts, it might be possible to read the basics of the following chapter and then directly to chapter 7.

2 Description of dual concurrent branching

2.1 Basic terminology and central theorem

In the following we examine a problem P, which can be partially represented as a minimal linear problem P_l. The desription of the entire problem needs some additional case differentiations. It should be remarked that the set of MILP-instances is a real subset of this problem class. The linear problem P_l has a set $\{y_m\}$ of inequalities in variables $\{x_n\}$. Without loss of the generality we

assume that all $\{y_m\}$ are $\geq$-inequalities. We'll argue later why equations may be excluded from the scope. Furthermore we expect that the inequalities named by y_m represent all bounds to the x_n.
Since the term branching has been used in LP-terminology in quite a lot of places with an emphasis on really creating two problems out of one problem two problems, a new terminology will be introduced. We use the terms of cases and files instead. A case $C_{i,j}$ will stand for the evaluation of one possibility j of a case differentiation i, the sum of the cases $C_{i,j}$ make together a file F_i, which will be in other words the case differentiation. We'll come to a more precise definition in a moment. We shall examine when and how the files can be combined to get a larger lower limit for the optimal solution.

Therefore we'll start from the dual point of view, so ω will be considered as the objective function of the dual problem. To simplify the notification, we state that the indices $\{m\}$ and $\{n\}$ have empty intersection. Via defining the index $\{r\}$ as the union of both we get something that makes all formulation much easier.

The dual solution space of P_l will be noted as V_0. Furthermore we chose an arbitrary $y \in V_0$. By looking at one case j of the case differentiation i the we call the dual solution space $V_{i,j}$. But we'll restrict this solution space by $\{\omega_{i,j} \geq \omega(y)\}$ to get a polytope $P_{i,j}$. Now let the case $C_{i,j}$ be the set $\{c_{i,j} |\ y + c_{i,j} \in P_{i,j}\}$. So $C_{i,j}$ is like $P_{i,j}$, but the fixed starting point y is the new zero vector. Our objective function $\omega_{i,j}$ can easily be expanded to $C_{i,j}$ in a natural way just by setting it to $\omega_{i,j}(c_{i,j})$ with $c \in C_{i,j}$, which is the same as $\omega_{i,j}(y + c_{i,j}) - \omega_{i,j}(y)$ due to the linearity of $\omega_{i,j}$.

Definition 2.1 *Choose $c_{i,j} \in C_{i,j}$ and define $\Delta^m_{i,j}(c_{i,j}) = y_m - (y + c_{i,j})_m = -(c_{i,j})_m$ as the change of the m-th dual variable. Also define $\Delta^n_{i,j}(c_{i,j}) = r_n(y) - r_x(y + c_{i,j})$, where r_n shall be the reduced cost of x_n. By this we also define $\Delta^r_{i,j}$.*

We make a break at this point and use a very easy example to illustrate the termns:

$$y_1 : x_1 + x_2 \geq 1 \text{ and } y_2 : x_2 + x_3 \geq 1 \text{ and } y_3 : x_3 + x_1 \geq 1$$

$$y_4 : x_4 + x_5 \geq 1 \text{ and } y_5 : x_5 + x_6 \geq 1 \text{ and } y_6 : x_6 + x_4 \geq 1$$

Minimize $\omega(x) = x_1 + x_2 + x_3 + x_4 + x_5 + x_6$ and all variables have to be integer.
Just by viewing at it you immediately see that the otimal solution is $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0.5$ with objective 3. But when integers are required the objective has to be at least 4.

Transformed to the dual space, we get:

$$x_1 : y_1 + y_3 \leq 1 \text{ and } x_2 : y_1 + y_2 \leq 1 \text{ and } x_3 : y_2 + y_3 \leq 1$$

$$x_4 : y_4 + y_6 \leq 1 \text{ and } x_5 : y_4 + y_5 \leq 1 \text{ and } x_6 : y_5 + y_6 \leq 1$$

Maximize $\omega(x) = y_1 + y_2 + y_3 + y_4 + y_5 + y_6$ The optimal solution is $y_1 = y_2 = y_3 = y_4 = y_5 = y_6 = 0.5$, we chose this point as the fixed starting point y. This point has for the inequalities x_1 to x_6 only reduced costs 0.

We now consider the cases $y_{1,1}^{\text{add}} : x_1 = 0$ or $y_{1,2}^{\text{add}} : x_1 = 1$. For the first case we get the following dual inequalities, which describes $V_{1,1}$:

$$x_1 : y_1 + y_3 + y_{1,1}^{\text{add}} \leq 1 \text{ and } x_2 : y_2 + y_3 \leq 1 \text{ and } x_3 : y_3 + y_1 \leq 1$$

$$x_4 : y_4 + y_5 \leq 1 \text{ and } x_5 : y_5 + y_6 \leq 1 \text{ and } x_6 : y_6 + y_4 \leq 1$$

We also look at optimal solution $\hat{y}$ of the dual problem with $y_{1,1}^{\text{add}} : x_1 = 0$ is $y_{1,1}^{\text{add}} = -1$ and $y_1 = y_3 = 1$ and $y_2 = 0$ and $y_4 = y_5 = y_6 = 0.5$. The value of the vector of the reduced costs for this dual point is still 0.

For the vector $c_{1,1}$ we see that its value is $(0.5, -0.5, 0.5, 0, 0, 0, -1)$, where the last coordinate represents the additional dual variable and $\omega(c_{1,1}) = \omega(\hat{y}) - \omega(y) = 3.5 - 3 = 0.5$. So $\Delta(c_{1,1})$ has the value $((-0.5, 0.5, -0.5, 0, 0, 0)(0, 0, 0, 0, 0, 0))$.

After this illustration we continue with the definition of the basic termns: It is important to state that the $\Delta_{i,j}^n(c_{i,j})$ can be calculated by the $\Delta_{i,j}^m(c_{i,j})$ with additional information of the value of the new dual variable(s) $y + c_{i,j}$. If we have equalities as conditions, we'll see that those dual values give no interesting Δ^m-values, but the Δ^n-values can be calculated with the help of these values. We also define the vector $l = (y, r_n(y))$, this vector has coordinates in $\{r\}$.
As $y + c_{i,j}$ is a dual solution, all coordinates y_m most be positive or null. The same holds for the linear function r_n, the value $r_n(y + c_{i,j})$ must be positive or null. Putting these facts together, we get the following remark, which already has the structure of our main statement 2.6:

Remark 2.2

$$\Delta_{i,j}^r(c_{i,j}) \leq l_r$$

For our example this vector l has the value $((0.5, 0.5, 0.5, 0.5, 0.5, 0.5)(0, 0, 0, 0, 0, 0))$.

For later purposes we investigate the property of linearity of the $\Delta_{i,j}^m(c_{i,j})$, the $\Delta_{i,j}^n(c_{i,j})$ and of the $\Delta_{i,j}^r(c_{i,j})$. We easily see that everything is linear:

Remark 2.3

$$\begin{array}{lcl} \Delta_{i,j}^m(\lambda c_{i,j}) & = & \lambda \Delta_{i,j}^m(c_{i,j}) \\ \Delta_{i,j}^m(c_{i,j} + d_{i,j}) & = & \Delta_{i,j}^m(c_{i,j}) + \Delta_{i,j}^m(d_{i,j}) \\ \Delta_{i,j}^n(\lambda c_{i,j}) & = & \lambda \Delta_{i,j}^n(c_{i,j}) \\ \Delta_{i,j}^n(c_{i,j} + d_{i,j}) & = & \Delta_{i,j}^n(c_{i,j}) + \Delta_{i,j}^n(d_{i,j}) \end{array}$$

In 2.2 we call those inequalities, where the right hand side is greater 0 the main inequalities.

Remark 2.4 *If $c_{i,j}$ relates to an optimal solution of $P_{i,j}$, then one main inequality is sharp.*

To see this we assume the contrary. We introduce ϵ as a very small and positive number. We consider $y + (1+\epsilon)c_{i,j}$, which has higher objective value and the inequalities in 2.2 still hold. As those inequalities make the dual variable related to the inequalities of $P_{i,j}$ positive and the dual inequalities of $P_{i,j}$ true. We conclude that $(1+\epsilon)c_{i,j}$ is in $C_{i,j}$. This is a contradiction to the optimality of $c_{i,j}$. So one of the main inequalities must be sharp. The other non-main inequalities are in fact trivial, since the values here for y itself are already sharp, so the Δ-values must be negative or null.

The next step of our thoughts is to go from a case to the case differentiations, which will be named files as announced. Let $F_i = \bigoplus_j C_{i,j}$ be a file and we take an element $f_i = \oplus_j c_{i,j}$ out of our construct F_i. We further define:

$$\omega_i(f_i) = \min_j(\omega_{i,j}(c_{i,j}))$$
$$\Delta_i^r(f_i) = \max_j(\Delta_{i,j}^r(c_{i,j}))$$

The delta represents the largest differences of the changes within a case differentiation (file) of the dual variables and the dual inequalities.

For the case of $x_1 = 1$ in our easy example, the we again choose the optimal point, which is: $y_{1,2}^{\text{add}} = 1$ and $y_1 = y_3 = 0$ and $y_2 = 1$ and $y_4 = y_5 = y_6 = 0.5$ and still reduced costs of 0. So we get for $c_{1,2} = (-0.5, 0.5, -0.5, 0, 0, 0, 1)$ with objective 0.5. Finally $\Delta(c_{1,2})$ has the value $((0.5, -0.5, 0.5, 0, 0, 0)(0, 0, 0, 0, 0, 0))$.

So we get that $f_1 = ((0.5, -0.5, 0.5, 0, 0, 0)(-0.5, 0.5, -0.5, 0, 0, 0))$ and $\Delta(f_1) = ((0.5, 0.5, 0.5, 0, 0, 0)(0, 0, 0, 0, 0, 0))$ and finally $\omega(f_1) = 0.5$. Another file f_2 could be for $x_2 = 0$ or $x_2 = 1$, when choosing again in bothes cases the optimal point you again get: $\Delta(f_2) = ((0.5, 0.5, 0.5, 0, 0, 0)(0, 0, 0, 0, 0, 0))$ and $\omega(f_2) = 0.5$.
Considering the file f_3 as the cases $x_4 = 0$ or $x_4 = 1$, we pick up again the optimal points for each solution and receive $\Delta(f_3) = ((0, 0, 0, 0.5, 0.5, 0.5)(0, 0, 0, 0, 0, 0))$ and $\omega(f_3) = 0.5$.

Based on 2.3 we get the following equalities and inequalities.

Remark 2.5

$$\begin{array}{lcl} \Delta_i^r(\lambda f_i) & = & \lambda\Delta_i^r(f_i) \\ \Delta_i^r(f_i + g_i) & \leq & \Delta_i^r(f_i) + \Delta_i^r(g_i) \\ \omega_i(\lambda f_i) & = & \lambda\omega_i(f_i) \\ \omega_i(f_i + g_i) & \geq & \omega_i(f_i) + \omega_i(g_i) \end{array}$$

We conclude that the deltas are still convex and ω_i is concave.
Now we build an even more complex space $\mathcal{P}$, which will be the sum of all files and our final object. This space represents parallel files.

$$\mathcal{P} = \bigoplus_i F_i$$

with the following functions for $p \in \mathcal{P}$:

$$\Delta^r_{\mathcal{P}}(p) = \sum_i \Delta^r_i(f_i)$$

$$\omega_{\mathcal{P}}(p) = \omega(y) + \sum_i \omega_i(f_i)$$

Now let $p \in \mathcal{P}$, then we have chosen in all cases of all files a solution vector. Remember that we are always talking about dual solutions and variables. If we chose for each file F_i a case $C_{i,j(i)}$, then we have a new problem $\hat{P}_l$, which is in fact P_l together with the inequalities from all $V_{i,j(i)}$. For this we can calculate a solution $\hat{y}$ by the means of the Δ: If we look at $\hat{y}_m$, then the value is:

$$\hat{y}_m = y_m - (\sum_i \Delta^m_{i,j(i)}(c_{i,j(i)})) \geq y_m - \sum_i \Delta^m_i(f_i) = y_m - \Delta^m_{\mathcal{P}}(p)$$

So by $\Delta^m_{\mathcal{P}}(p) \leq y_m$ we can make sure, that the dual variable $\hat{y}_m$ is positive. Since this is not important for equations, we had only considered those indices m that represent inequalities. Notice also that the condition is independent of our choice $j(i)$. It should be noted that the additional variables of the cases can clearly be handled as independant of each other. So for $V_{i,j(i)}$ all additional variables only occur once.

For the validity of the n-th dual inequality we got something similar:

$$\begin{array}{lcll} \Delta^n_{\mathcal{P}}(p) & \leq & r_n(y) & \Rightarrow \\ \sum_i \Delta^n_{i,j(i)}(c_{i,j(i)}) & \leq & r_n(y) & \Leftrightarrow \\ 0 & \leq & r_n(y) & \end{array}$$

Keep in mind, that also $r_n(\hat{y})$ can be calculated by the $\Delta^m_{i,j(i)}(c_{i,j(i)})$ with the help of the additional dual variables of all chosen cases. This is the same as the $\Delta^n_{i,j}(c_{i,j})$ could be derived from the $\Delta^m_{i,j}(c_{i,j})$, as we have seen before.

The objective value of $\hat{y}$ is $\omega(y) + \sum_i \omega_i(c_{i,j(i)}) \geq \omega_{\mathcal{P}}(p)$. Putting these thoughts together we get our central statement.

Theorem 2.6 (Central theorem) *If for all r holds that $\Delta^r_{\mathcal{P}}(p) \leq l_r$, then the original problem P must have an optimal solution that is greater than $\omega_{\mathcal{P}}(p)$.*

Via this theorem we see that the vector (f_1, f_3) gives a lower limit of 4. Also it is clear that it can't be applied to (f_1, f_2)

2.2 Building the little combining LP

The last statement seems to be rather abstract, but by an easy trick, we'll get two different forms, that can be used in an algorithm. To get the first we just substitute f_i by $\lambda_i f_i$ with $\lambda_i \geq 0$. As the deltas and the objectives ω_i are linear on a scalar (2.5), we get the main result of this section:

Theorem 2.7 (Central theorem - simple form) *The ability to combine case differentiations can be assured by the following inequalities:*

$$\sum_i \lambda_i \Delta_i^r(f_i) \leq l_r$$

The new lower limit is $\sum_i \lambda_i \omega_i(f_i) + \omega(y)$.

So by solving this LP-instance in λ_i we get a better lower limit for our problem P.

For a good example of the usage of this theorem we refer to 4.

By looking at all $P_{i,j}$ all values in this LP-instance can be calculated, first the $(y + c_{i,j})_m$ and $r_n(y + c_{i,j})$, secondly $\Delta_{i,j}^m(c_{i,j})$, $\Delta_{i,j}^n(c_{i,j})$, $\omega(c_{i,j})$ and lastly the $\Delta_i^m(f_i)$, $\Delta_i^n(f_i)$ and the $\omega_i(f_i)$.

By the definition of ω_i it is natural to choose the $c_{i,j}$ in such a way that for all j the equation $\omega_i(f_i) = \omega_{i,j}(c_{i,j})$ holds. This can be achieved by substituing $c_{i,j}$ with $\omega_i(f_i)\omega(c_{i,j})^{-1}c_{i,j}$. This is well-defined for our purposes, because when $\omega(c_{i,j}) = 0$ then we get no progress on the objective function of P from this case differentiation. By this substitution in 2.7, the objective function $\omega_i(f_i)$ remains the same, but normally the Δ-values will decrease, leading to higher values when using the practical form of the central theorem. We call this *normalization*.

As a next step we want to generalize 2.7. We substitute in 2.6 f_i by $\sum \lambda_{i,k} f_{i,k}$ and use the convexity of Δ_i^r (2.5) to get the second sum in the upcoming theorem 2.8.

Theorem 2.8 (Central theorem - more complex form) *The ability to combine case differentiations can be assured by the following inequalities:*

$$\sum_{i,k} \lambda_{i,k} \Delta_i^r(f_i^k) \leq l_r$$

The new lower limit is $\sum_{i,k} \lambda_{i,k} \omega_i(f_i^k) + \omega(y)$.

Although this formulation seems to be much stronger than the simple form, this is not really the case. Looking at the sum we see that the $\lambda_{i,k}$ can be

created in 2.7 by choosing $F_{i_1,j} = F_{i_2,j}$ for $i_1 \neq i_2$. This is possible because it was never stated that we made a case differentiation only once.
But by the construction of 2.8 we see something different: If some λ_{i,k_1} and λ_{i,k_2} with $k_1 \neq k_2$ are non-null for an optimal solution of the resulting LP-instance in 2.8, then we can find better values by setting $f_{i,\hat{k}} = \lambda_{i,k_1} f_{i,k_1} + \lambda_{i,k_2} f_{i,k_2}$. So by generating new columns we can sometimes improve the bound for P.

We have presented in this section a theory on a problem, which can only partly described by a LP. But in truth we studied the dual LP, where the problem is described too sharp and can be weakened by case differentiations. As 2.7 can be weakened by natural case differentiation by fixing for one i and one j $\Delta_i^r = \Delta_{i,j}^r$, we could use the entire theory on it.

We will sketch one manual example later at 4. Even if this looks interesting on the first glance, we'll see in the last chapter 8 that this iteration could have been reached just by a wider case differentiation already in the first combination of files.

Although the mathematical formulation to combine case differentiations (files) has been explained broadly in this section, some details are still not covered. The problem is that those details might be not too easy to attack at all. When you think of a fast implementation of this idea you want to have an effective, numerically stable and fast algorithm to find good elements $c_{i,j} \in C_{i,j}$, where most $c_{i,j}$ are zero. From these you get good f_i for a given solution y. We'll see later in 3 that the normal approach to use optimal solutions of the $P_{i,j}$ leads in some examples to problems. So later in 5.2 and 5.3 we will attack these problems by using non-optimal $y + c_{i,j}$ even before normalization, where most $\Delta_{i,j}^m(c_{i,j})$ and $\Delta_{i,j}^n(c_{i,j})$ should be zero.

2.3 Rough description of the huge combining LP

In this section we will follow again the made definitions and results and sketch a mathematical satisfactory formulation of the theory.
We had started with one solution $y \in V_0$. For each case of a file we have also a $y_{i,j} \in V_{i,j}$. Notice that the reduced costs of one variable of y^0 (and $y_{i,j}$) are by definition just a linear equation dependent of the $(y^0)_n$ ($(y_{i,j})_n$). So we define as in 2.1 the $\Delta_{i,j}^r$ as variables which are calculated by linear equations from y^0 and $y_{i,j}$, the same holds for the $\omega_{i,j}$, which are also linear dependent on y^0 and $y_{i,j}$.
Via the restrictions $\Delta_i^r \geq \Delta_{i,j}^r$ and $\omega_i \leq \omega_{i,j}$ for all j we have defined Δ_i^r and ω_i as linear inequalities. Like before we define $\Delta_{\mathcal{P}}^r$ as a sum of the Δ_i^r and $\omega_{\mathcal{P}}$ as the sum of the ω_i plus the objective ω of P_l, which is also a linear term of y^0. Via using the restrictions of 2.6 and setting the objective to $\omega_{\mathcal{P}}$ we have defined now a huge LP P_{comb}. We can now postulate a theorem, which describes the problem of doing case differentiations in parallel in a mathematically satisfactory way:

Theorem 2.9 (Central theorem - complete form) *Each solution of P_{comb} represents a lower bound of P.*

The optimal value is the optimal lower bound possible via our combining technique.
We restricted ourselves to writing all inequalities explicitly down here, as the huge amount of indices for each variable might only be confusing.

But it should be noticed that the definitions in 2.7, 2.8 and 6.1 are more tightened inequalities systems, which are also less complex in comparison with this huge inequality system.

In the last section 8 of this paper we will really write down the LP and discuss its properties. Anyhow this LP is very complex, but in the author eyes it is playing a central role for studying MILP in general.

3 Implementation with usage of optimal solutions of the subproblems

Putting the thoughts from the previous section together you get the following algorithm described as pseudo-code to get higher objective values of a MILP-instance with only 0-1 variables:

derive P_l from given MILP-instance
load P_l into LP-solver
solve P_l and save one optimal solution x and the fitting dual solution y
for all i, where $(x)_i$ is not integer **do**
  case $j = 1$:
  add inequality $x_i \leq 0$ to P_l (so getting $P_{i,1}$)
  solve this LP by usage of the old dual solution y and get $y + c_{i,1}$ as dual solution
  calculate all $\Delta^r_{i,1}(c_{i,1})$- {as defined in 2.1}
  case $j = 2$:
  add inequality $x_i \geq 1$ to P_l (so getting $P_{i,2}$)
  solve this LP by usage of the old dual solution y and get $y + c_{i,2}$ as dual solution
  calculate all $\Delta^r_{i,2}(c_{i,2})$ - {as defined in 2.1}
  $\omega_i(f_i) = \min(\omega_{i,1}(c_{i,1}), \omega_{i,2}(c_{i,2}))$
  if $\omega_i(f_i) > 0$ **then**
    mark i
    if Normalization trick is wanted **then**
      if $\omega_{i,1}(c_{i,1}) \leq \omega_{i,2}(c_{i,2})$ **then**
        for all r: $\Delta^r_{i,2}(c_{i,2}) = \omega_{i,1}(c_{i,1})(\omega_{i,2}(c_{i,2}))^{-1}\Delta^r_{i,2}(c_{i,2})$
      else
        for all r: $\Delta^r_{i,1}(c_{i,1}) = \omega_{i,2}(c_{i,2})(\omega_{i,1}(c_{i,1}))^{-1}\Delta^r_{i,1}(c_{i,1})$

21: **end if**
22: **end if**
23: for all r: $\Delta_i^m(f_i) = \max(\Delta_{i,1}^r(c_{i,1}), \Delta_{i,2}^r(c_{i,2}))$
24: **end if**
25: **end for**
26: build new LP-instance R with all marked i in variables λ_i as described in 2.7
27: solve R

In the first implementation I was not able to use the old solution in lines 7 and 11 effectively. This has some impact because of the degeneration of the optimal dual solution in most of the prominent problems.
To understand this let's consider you have chosen an optimal y with $(y)_m > 0$ for a problem P. But also an optimal $\bar{y}$ exists with $(\bar{y})_m = 0$. Now for all i one $c_{i,j}$ could exist where $(y + c_{i,j})_m = 0$. This leads to the situation that no file could be combined ensured by the inequality of R, which deals with the fact that dual variables for inequalities should be positive. But if you had chosen the other $\bar{y}$, you would have less problems. As a side-remark it should be noticed that in this situation also all $P_{i,j}$ then have a degenerate dual solution space.
The approach to use the old solution y as a starting point for solving $P_{i,j}$ has two benefits: First the optimal solution should be found faster numerically and secondly normally when dealing with degeneracy the above described effect should happen less often.
The above algorithm has been implemented with the Open-Source package glpk. In this program all MILP are transformed to be Minimum-problems by exchanging the sign of the objective. The problem library MIPLIB2003 [5] has been processed partially getting the results on the following page.

In the given table the column *Branches* measures the number of variables, where branching took place. The actual number of calculated LPs is 2 times more plus the initial LP and the combining LP. The column *Degree* is equal to $\sum_i \lambda_i$ of the optimal value of the combining LP. It gives an idea how much branches can be used at the same time but also in an effective way. So we measure both in seconds. Furthermore also the total time for all calculation is presented.
The given table only includes those instances, where the program finished within 1 hour. Furthermore for some instances no advantage at all was made, because at no branch there was an increase in both nodes at all. For more investigations these problems might be put out of scope. On the other hand for the instance tr12-30 a quite high lower bound was reached: Starting from 14210 the bound 79695 is reached which is much nearer to the real value at 130596.

Instance	Pure LP	Bound Inc	Branches	Degree	Normal	Dual	Total
10teams	917.00	0.00	159	0.00	3	1	11
a1c1s1	997.53	1195.33	173	53.92	5	1	24
aflow30a	983.17	14.28	31	5.19	1	0	2
aflow40b	1005.66	7.16	38	1.80	3	2	17
air04	55535.44	84.61	292	1.00	59	21	1040
air05	25877.61	72.54	223	1.00	28	5	329
arki001	7579599.81	126.65	81	6.63	5	2	12
cap6000	-2451537.33	0.00	2	0.00	12	5	18
danoint	62.64	0.05	34	1.00	0	1	3
disctom	-5000.00	0.00	251	0.00	69	0	129
fiber	156082.52	15734.31	47	6.90	1	0	3
fixnet6	1200.88	210.51	60	21.33	1	0	2
gesa2	25476489.68	81043.25	58	35.91	2	1	5
gesa2-o	25476489.68	81891.56	73	36.70	1	0	4
glass4	800002400.00	0.00	72	0.00	1	0	1
harp2	-74353341.50	0.00	30	0.00	3	1	6
liu	346.00	214.00	536	1.00	2	1	16
manna81	-13297.00	0.00	872	0.00	10	1	92
markshare1	0.00	0.00	6	0.00	0	0	0
markshare2	0.00	0.00	7	0.00	0	0	0
mas74	10482.80	42.52	12	1.19	1	0	1
mas76	38893.90	24.86	11	1.62	0	0	0
misc07	1415.00	0.00	31	0.00	1	0	1
mkc	-611.85	0.00	105	0.00	5	0	16
mod011	-62121982.55	0.00	16	0.00	13	1	16
modglob	20430947.62	69955.22	29	8.31	0	0	0
mzzv11	-22945.24	0.00	836	0.00	68	1	323
mzzv42z	-21623.00	0.00	676	0.00	48	1	278
net12	17.25	11.40	429	1.30	27	89	3115
noswot	-43.00	0.00	28	0.00	1	0	1
nsrand-ipx	48880.00	0.00	67	0.00	37	2	61
opt1217	-20.02	0.00	29	0.00	1	0	1
p2756	2688.75	10.20	30	2.00	3	0	4
pk1	0.00	0.00	15	0.00	0	0	0
pp08a	2748.35	762.82	51	11.41	0	0	0
pp08aCUTS	5480.61	166.85	46	6.47	1	0	1
protfold	-41.96	0.00	449	0.00	7	1	34
qiu	-931.64	0.00	36	0.00	1	1	3
roll3000	11097.13	5.44	214	4.32	6	1	36
rout	981.86	2.34	35	1.00	0	1	1
set1ch	32007.73	3904.90	138	64.56	1	0	2
seymour	403.85	1.50	632	3.30	30	4	291
sp97ar	652560391.11	241502.97	194	2.00	89	12	522
swath	334.50	0.40	45	5.71	5	1	19
timtab1	28694.00	137970.93	136	16.46	0	0	1
timtab2	83592.00	106311.17	233	27.02	0	1	4
tr12-30	14210.43	65484.48	348	322.01	1	0	8
vpm2	9.89	0.48	31	7.41	1	0	1

4 Another example of the presented technology

$$
\begin{aligned}
x_1 + x_2 + x_3 &\leq 1\\
x_2 + x_3 + x_4 &\leq 1\\
x_3 + x_4 + x_5 &\leq 1\\
x_4 + x_5 + x_1 &\leq 1\\
x_5 + x_1 + x_2 &\leq 1
\end{aligned}
$$

Maximize $\omega(x) = x_1 + x_2 + x_3 + x_4 + x_5$ and all variables have to be integer.

Let $i \in \{1; \ldots; 5\}$, the optimal dual solution of the LP is simply $y_i = \frac{1}{3}$ with $\omega = \frac{5}{3}$. The optimal solution of the MILP has objective of 1. For example we branch on the two cases $x_1 = 0$ and $x_1 \geq 1$. We get then the following dual solutions:

$$
\begin{array}{llllllll}
x_1 = 0: & y = & (0, & 0.5, & 0.5, & 0, & 0.5) & \text{with } \omega = 1.5\\
x_1 = 1: & y = & (1, & 0, & 0 & 1, & 0) & \text{with } \omega = 1\\
x_1 = 1: & y = & (0.5, & 0.25, & 0.25, & 0.5 & ,0.25) & \text{with } \omega = 1.5 \text{(Normalization!)}
\end{array}
$$

Naturally the theory can also be applied to maximum problems. So we get for Δ_1:

$$\Delta_1 = (\frac{1}{3}, \frac{1}{12}, \frac{1}{12}, \frac{1}{3}, \frac{1}{12}); \text{ with } \omega = \frac{1}{6}$$

Via using the symmetry of the problem we get the combination of the files the following LP:

$$
\begin{array}{rcrcrcrcrcl}
\frac{1}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
\frac{1}{12}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
\frac{1}{12}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3}\\
\frac{1}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
\frac{1}{12}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3}
\end{array}
$$

With objective $\frac{1}{6}\lambda_1 + \frac{1}{6}\lambda_2 + \frac{1}{6}\lambda_3 + \frac{1}{6}\lambda_4 + \frac{1}{6}\lambda_5$.
The optimal solution is $\lambda_i = \frac{4}{11}$ and objective is $\frac{10}{33}$. So that we have shown that the maximum in our original MILP is less or equal $\frac{5}{3} - \frac{10}{33}$.

At this point it is again possible to make a case differentiation on $x_1 = 0$ or $x_1 \geq 1$. If we assume $x_1 = 0$ the above LP would have the following form:

$$
\begin{array}{rcrcrcrcrcl}
\frac{1}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
-\frac{1}{6}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
-\frac{1}{6}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3}\\
\frac{1}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3}\\
-\frac{1}{6}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3}
\end{array}
$$

And for $x_1 \geq 1$:

$$\begin{array}{rcrcrcrcrcr}
-\frac{2}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3} \\
\frac{1}{3}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3} \\
-\frac{2}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{3}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3} \\
\frac{1}{3}\lambda_1 & + & \frac{1}{12}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{3}\lambda_4 & + & \frac{1}{12}\lambda_5 & \leq & \frac{1}{3} \\
-\frac{2}{3}\lambda_1 & + & \frac{1}{3}\lambda_2 & + & \frac{1}{12}\lambda_3 & + & \frac{1}{12}\lambda_4 & + & \frac{1}{3}\lambda_5 & \leq & \frac{1}{3}
\end{array}$$

We'll stop the calculation at this point. We could now calculate some Δ_1 via checking the differences for the resulting normal variables λ_i and build a new LP, which would represent the possibility of combining the changes of λ_i when doing the case differentiations. By this we would again reach better upper limit for the original problem.

It is possible to iterate this method, but some manual calculations have shown that the real lower limit will never be reached in this way.

The examples shows that non-trivial combining is possible, and that the method can be iterated in a surprising way. But it also shows some limits. The above MILP is easily solved by doing the 4 case differentiations on x_1 and x_2. Furthermore it is even possible to make another case differentiation on one inequality. It is clear via the first inequality that $x_1 = 1$ or $x_2 = 1$ or $x_3 = 1$ or $x_1 = x_2 = x_3 = 0$ holds. Via this case differentiation it is seen most quickly that the optimal value of the MILP is 1.

5 Effectiveness for finding good dual values in the branching LPs

5.1 Excursion: Searching for integrity

Only loosely connected to the rest of the paper we now investigate those MILPs and the derived LPs which have a non-degenerate optimal solution space. As for all branching investigations especially in this paper the number of non-integer variables, which are supposed to be integer, should be as small as possible to reduce the running time of an implementation.
Therefor we assume that we have an optimal solution vector x_0. We just freeze the objective function to the optimal value, so getting an additional equality. We set additional bounds on all integer variables x_n via $[(x_0)_n] \leq x_n \leq [(x_0)_n] + 1$. This is a good valid definition also for MILPs which are not binary problems.
Finally we now define for all integer variables n the objective of the minimization problem to enhance the variables to become integer.

$$c_n = 1 - 2([(x_0)_n] - (x_0)_n)$$

The new vector x_1 is now got by solving this LP to optimality. We now calculate again new c_n in the described manner so that we have an iterative process.

With this definition we have a good tool which gives numbers which are almost integers a good reason to become integer not hindering others in this process to give up integrity.

Notice that the choice of the c_n was done by experiments. It cannot be reasoned yet, why this choice was in the experiment superior to other approaches. Also the convergence of the method has only investigated by experiments.

Clearly this presented idea was motivated by the feasibility pump [2] to generate integer solutions. We present it also here because the following method was developed in spirit of this easy algorithm.

5.2 Measurement of good dual values

We will again concentrate on theorem 2.7. Looking at the inequality there you see that each file eats up certain inequalities (dual values) or variables (reduced costs). So to find good values, you have to search for files and hereby for dual variables who eat less of our stock but still give a good improvement in the objective function. First we have to define what it is the meaning is of eating up the stock of inequalities and variables.
Suppose again you have made a case j of case differentiation i with a better dual variable set. Some of the $\Delta_{i,j}^r$ might be negative, but when the file is glued together by maximizing we suspect that the value Δ_i^r will be positive. Anyhow even if it is negative, quite likely no other case differentiation will need the negativeness. So we have argued to measure all negative $\Delta_{i,j}^r$ as 0. As a general approach measure distance to the starting point y^0 we can now define:

$$D = \sum_r d_r \max(0, \Delta_{i,j}^r)$$

We also needed $D \geq 0$ to make the algorithm below work.

We leave out the problem of setting the d_r values, but first use this definition to get better dual values. Therefor we create more artificial variables z_r in the dual space. Via $z \geq 0$ and $z \geq \Delta_{i,j}^r$ and $D = \sum_r z_r$ we fullfill the above definition. The idea is now to subtract D from the objective ω in that way the optimal value of the LP created by the case j of the case differentiation will have the same objective in our new artificial LP as y^0. Let $D_{i,j}^0$ be the difference of this solution and $\omega_{i,j}$ the increase of the objective. Then we set:

$$\omega_{i,j}^{\text{adj}} = \omega - \frac{D\omega_{i,j}}{D_{i,j}^0}$$

So we have found an objective with the desired property.

Via our definitions we have assured that D is always positive. When the new LP is now solved to optimality getting the point $y_{i,j}^{\text{new}}$, it is therefor clear that

its optimal value is in the polytope $P_{i,j}$. Furthermore the following can easily be proved:

$$\frac{\omega_{i,j}^{\text{new}} - \omega_0}{D_{i,j}^{\text{new}}} \geq \frac{\omega_{i,j} - \omega_0}{D_{i,j}}$$

So in terms of efficiency of eating up the stock the new point is better or equal than the first optimal point. When it is equal, then the space $P_{i,j}$ might be often one dimensional. But additionally by our definition we have not given up the wish for good objective gain.
Also this trick can easily be iterated, visible already by our definitions.
The algorithm can easily be enhanced so that it works on finding better values of files, but this generalizations will not be presented here. Also in an implementation one could use the values of the already manipulated cases of the case differentiation. When an inequality or a variable has been eaten up a bit the new case should have this meal for free.

We discussed efficient meals of inequalities and variables, but one crucial point of the receipt is still open: the definition of the d_r. If you define all $d_r = 1$, then the big l_r in 2.7 will get two much attention. Tiny l_r, which might always hinder the combining of the files, are overlooked.
So the natural choice is to set $d_r = \frac{1}{l_r}$ which will give all non-null inequalities and all variables with real reduced costs the same weight. Sadly this theoretically good approach led in the author implementations to numerical problems. Often the manipulated LP was bad conditioned. So the author suggests to use a lower limit like 0.01 for all d_r.
The author has implemented the above algorithm partially, but with not too good results. He didn't manage to use the old optimal solution in the software package glpk, so each manipulated LP had to be solved from scratch. This led to too long running times. He thinks also that this time increase is only partial because of some missing features of the used software. Using the theoretical good reasoned approach of this section might just be numerically too complex because of the sheer number of added constraints.
Later on we'll understand, that cuts can be generated via the differences of the dual values. These cuts could also tried to be optimized with this idea. When the problem is good conditioned it should be a good idea for the cut creation to demand just $d_r = 1$.

5.3 Finding the dual values quickly

In the preceding subsection we described a theory to find dual solutions with good objective which could be considered as near to the basic dual solution y^0. We did this via introducing variables, which measured the distance to the original. Another approach in finding good dual solutions and so files is to use additional inequalities. Depending on the aim this can result in files which fit better to each other or in dual solutions which can be calculated very quickly.

Suppose you have already made a branching with a file f_1. Then to combine a second branching with the first you just demand:

$$l_r - \Delta^r_{1,j} \geq \Delta^r_{2,j} \forall r, j$$

Speaking in terms of dual inequalities you get lower bounds for those variables y_m, which were nonzero in the basic dual solution. Furthermore restrictive dual inequalities which weren't in the solution vector y^0 become in general more restrictive. The benefit of this approach is that using those additional restrictions it is clear that the two branching can be fully combined. In terms of the theorem 2.7 this means that $\lambda_1 = \lambda_2 = 1$.
Naturally the idea can easily be iterated via demanding:

$$l_r - \sum_i \Delta^r_{i,j} \geq \Delta^r_{i+1,j} \forall r, j$$

At this point we make again a little excursion. Suppose that both files consist of two cases, which is the normal case for MILPs. You have done the case differentiations to combine these two cases, so you have solved $2 + 2 = 4$ calculations. But doing instead a case differentiation on the the 4 cases ($2 * 2 = 4$), which already enumerate all possible combinations, you would have the same calculation time. But you will have a least a larger increase of the objective with these 4 cases than with combining the two case differentation. So for a clever implementation of these sketched algorithms the principle of parallel branching should not be followed too strictly. Doing all case differentiations on p cases might be interesting, when 2^p ist still comparable to $2p$.

But let's get back to additional restrictions for the dual inequalities. We start with a metaphor: Linear equations describe the nature. When a butterfly flies up in Brazil the emerging circulations won't normally influence the weather in Europe. Speaking in terms of LP an introduction of a new variable in a dual inequality has often only effect in those inequalities, which are strongly bound to the related inequalities. So it is striking a thought to define the neighborhood of a new variable, and to freeze all other variables, which are not in the neighborhood. This should have a big reduction of the running time as a result.

Clearly defining neighborhood by the graph of the inequality system or other means is a complex story. The definition of the neighborhood should also be dependent on the type of the MILP.
Suppose you have a good neighborhood definition. Then the technique of freezing most of the dual variables might also be an alternative to the strong branching method, which determines in a branch and cut framework the next variable to branch on as described in [7]. The strong branching relies on a good and steep implementation of the dual simplex, where you use the values of the objective after only some iteration of the Simplex algorithm. It should be noticed that such a steep dual Simplex algorithm is not a prerequisite of the algorithm. So my approach can be used in more simple LP-packages like glpk to do something

similar.

This chapter could be considered as metaphoric, anyhow the subject of this paper is to present the author's idea on the subject. The ideas seen in the context of the following chapters could be the basis for effective cut generations.

6 Application of the theory to produce cuts for the original MILP

When thinking of integrating in the look ahead term of the concurrent branching into an existing branch and cut framework, the dual combining inequality of 2.7 does not fit easily. It is striking that instead of that additional dual LP you would just like to have more restrictions in normal space instead. Generation of cuts should be the aim. We will see that this is possible in the end of the chapter.

We start at the point that we have an optimal dual solution y^0 with only one file f_1, which describes a case differentiation. We now use a new special form of theorem 2.9 . For this we freeze the f_i as linear factors of a scalar λ. Contrary to the special form 2.7 we really let y play the role of dual variables and do not fix it to l_r-values. So our variables are the vector y and the scalar λ. Transformed back via dual-dual correspondence this will give us more or less the normal inequalities and equalities plus one additional equality, which will be our cut. But let us stick to the details. We have the following dual inequalities for this special model:

Definition 6.1

$$x_m^{add} : \lambda \Delta_i^m(f_i) \leq y_m \forall m$$
$$x_n : \sum_m a_{m,n} y_m + \lambda \Delta_i^n(f_i) \leq c_n \forall n$$

Where (n) goes over all normal variables, (m) over all dual variables (inequalities + equalities), $a_{m,n}$ are the matrix coefficients of the LP and c_n are the coefficients of the normal objective. The objective function of this dual problem is $\sum_m b_m y_m + \lambda \omega(f_i)$, where the b_m are the right hand sides of the inequalities and equalities.

This dual problem can be transformed to normal space:

Remark 6.2

$$y_m : \sum_n a_{m,n} x_n - x_m^{add} \geq b_m \forall m$$
$$\lambda : \sum_m \Delta_i^m(f_i) x_m^{add} + \sum_n \Delta_i^n(f_i) x_n \geq \omega(f_i)$$

The objective is just $\sum_n c_n x_n$ as the normal objective. This looks already interesting, but prior to the final transformation to get a cut we must first prove that this system is valid for all integer values. This first proof is very indirect, anyhow the result for the increase of the objective is stronger than the direct prove in 7.1.

Before doing the proof we must first study the reuse of files for other basic solutions than the starting one. In 2.7 we had some files, which we tried to add to some basic solution. If we would use another solution with other l_r-values, we could naturally use the methodology also. Adding the files might still be possible. The only thing which might happen is that all λ_i in 2.7 have to be 0. The same holds if we have a stricter LP. Then the dual solution has only some more variables, but the original ones are still there.

Remark 6.3 *The lookup term via combining files can still be used to a stricter version of the starting normal LP. For incompatible problems it can only be reused, when the missing dual variables Δ_i^m of the LP, where the file should be applied, are negative or* 0.

This remark also clarifies the usage of the lookup terms for integration in branch and cut frameworks. The file info of a LP remains valid for all descendants and is normally invalid for other descendants of the root LP.
Consider you have an integer solution of the LP. Then it is clear that this integer solution is the only optimal solution of a version of the LP, which has been made stricter via adding more inequalities. This more restrictive LP is represented in the dual space by a loosened LP. We can still try to add our file in the dual space. Via this we get the special model 6.1 for the loosened dual inequality. For this model the optimal objective has to be identical to the dual LP and the normal LP. Otherwise we would prove that the optimal integer solution of the more strict LP has to have a bigger objective than the already existing integer solution, which is a contradiction.
The optimal dual solution of our loosened LP 6.1 in the dual space is a solution of a more restrictive LP 6.2 than the original one. So we have found an optimal normal solution, which also holds for the more restrictive inequalities. As we have said that the original solution was the only optimal solution, it must be identical to the new one. So the original solution has to fulfill 6.2.
So all integer solutions fulfill it.

As a final step we can state that the normal inequality system is equivalent to:

$$y_m : \sum_n a_{m,n} x_n \geq b_m \forall m$$

$$\lambda : \sum_m (\Delta_i^m(f_i) \sum_n b_n - a_{m,n} x_n) + \sum_n \Delta_i^n(f_i) x_n \geq \omega(f_i)$$

Or written with slack variables $s_m = \sum_n b_n - a_{m,n} x_n$:

$$y_m : \sum_n a_{m,n} x_n \geq b_m \forall m$$

$$\lambda : \sum_m \Delta_i^m(f_i) s_m + \sum_n \Delta_i^n(f_i) x_n \geq \omega(f_i)$$

The cut λ in its last form is surprisingly short and that is where we aimed to go. The dual LP 6.1 has at least an increase of the value of the optimal solution of $\omega(f_i)$. So the same holds for its dual which is equivalent to the last inequality.

Theorem 6.4 (Generation of branching cut) *The following inequality is true for all integer solutions:*

$$\lambda : \sum_m \Delta_i^m(f_i) s_m + \sum_n \Delta_i^n(f_i) x_n \geq \omega(f_i)$$

The increase of the objective by adding one cut of this kind is at least $\omega(f_i)$.

6.1 Thoughts about the new cuts

First we apply this cut to the problem in 4 and we get:

$$\frac{1}{3}s_1 + \frac{1}{12}s_2 + \frac{1}{12}s_3 + \frac{1}{3}s_4 + \frac{1}{12}s_5 \geq \frac{1}{6}$$

$$\frac{9}{12}x_1 + \frac{6}{12}x_2 + \frac{6}{12}x_3 + \frac{6}{12}x_4 + \frac{6}{12}x_5 \leq \frac{11}{12} - \frac{5}{6}$$

$$\frac{3}{4}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_3 + \frac{1}{2}x_4 + \frac{1}{2}x_5 \leq \frac{3}{4}$$

When applying $x_1 = 0$ as case in the original problem, you get the solution vector $(0, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{2})$. This solution is equalizing the above cut. And for the other case $x_1 = 1$ with solution vector $(1, 0, 0, 0, 0)$, this is also sharp.

If we would have chosen the file without normalization, we would have got:

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq \frac{3}{2}$$

At this cut $(0, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{2})$ is equalizing the cut, but $(1, 0, 0, 0, 0)$ is not. This could easily be investigated in a more abstract way. Anyhow we state: Normalization leads to sharper cuts, which is true in general.

Remark 6.5 *The defined class of the cuts are sharp, in the sense that it can be used to get a proof that an integer solution is the optimal one.*

For binary problem this is not difficult to understand. Just make a case differentiation over all cases. As we have a binary problem this is a finite number, so this one derived cut is already sufficient. In the general case, the following is true: If an optimal integer solution exists then you can find this by a finite case

differentiation. Then our method would in principle be capable to derive a cut to produce a proof.
The above statements can not be applied solve the MILP, as by making a case differentiation you already had a proof.
The derived cut will be similar or equal to the objective as above $x_1 + x_2 + x_3 + x_4 + x_5 \leq \frac{3}{2}$ was already the objective function, but not with the real optimal integer objective value.

We will now get a little more abstract. Consider you want to make a proof that an integer solution with objective ω_0 is an optimal one. By a big case differentiation you can produce one single cut, so that the best integer solution has to be almost ω_0. But the cut is already very similar to the objective function. So if you make a simple case differentiation after adding the cut, the objective will not increase in any branch at all. Thus the cut with coefficients similar to objective is irrelevant for the proof at all. This suggest the below expectation :

Remark 6.6 *Many little inequalities with few coefficients are better than big inequalities which are alike the objective.*

If you analyse the proof of the validity of the cut, things like parallelism of branching are not used at all. This could lead to the wrong conclusion that the whole dual theory of concurrent branching is redundant. In the dual space you can calculate which files to use, measure the files and so the cuts. In dual space you just have better control of what you do.

The author has studied a few other articles ([4] and [6]) about other cut classes. He found some similarities of disjunctive cuts with his idea. For instance in [4] the deepest cut is defined for a case differentiation. As this cut must be optimal, it must be very similar or even equal to the cut defined by optimal dual points in the underlying cases in our terms. We'll not try to find at this point a direct connection to those cuts via transforming CGLPs (Cut Generation Linear Programming) to dual. Instead in 7.3 we show something much stronger.

Also in the literature problems were mentioned with those cuts. The generated cuts had too big coefficients or were too complex or were useless in contrast to much easier schemas. Following our thoughts about effectiveness of cuts and good files in 5.2, these problems could also show up when using our cuts.

7 Application of the theory to cuts

In this chapter we demand that there are no reduced cuts, meaning that all dual inequalities are sharp via introducing variables for x_i. This is one of three equivalent models of presenting dual LP:

1. Only usage of equations, so in our cut generation inequality 6.4 we only have the reduced costs term.

2. Using inequalities but not equalizing the dual inequalities. Here we have both sums.

3. Sharp dual inequalities.

As all three are logical identical, it is a must that in our framework they produce the same cuts. Anyhow this is the case.

7.1 Adding inequalities in the dual view

We start with an inequality system $\{y_m\}$ with some objective ω and a dual point y', which doesn't have to be optimal. To this we add the inequality y_j. Handling the inequality y_j like a case in the preceding chapters, we get from the (non-)optimal dual solution y_j^0 some $\Delta^m(y_j^0)$ and a $\omega(y_j^0) - \omega(y')$. On the other hand we could have subtracted from the linear representation of the function $\omega(x)$ by the $\{y_m\}$ and y_j with values y_j^0 the other linear representation $\omega(x)$ by just $\{y_m\}$ with values y'. When $(y^0)_j$ is unequal 0 we get via this procedure a formula of the inequality y_j, which is by definition equal to the inequality from the usage of y_j as a file consisting only out of one case by our cuts of 6.4.

Theorem 7.1 (Recovering Inequalities by usage of Dual Deltas) *All inequalities can be transformed to the delta form.*

$$y_j : (y^0)_j^{-1} \sum_m \Delta^m(y_j^0) s_m \geq \omega(y_j^0) - \omega(y')$$

As an inequality is only defined up to a factor, we can demand $(y^0)_j = 1$.

At this point it is possible to give an easy direct proof of the validity of the inequality at 6.4. All the underlying inequalities of a case differentiation can be transformed to a form like in 7.1. Via doing all maximizations on the $\Delta^m(y_j^0)$ and minimization of $\omega(y_j^0) - \omega(y')$ all inequalities are weakened, so the cut 6.4 becomes valid for all cases. So it is a valid cut.

7.2 The relationship between our branching cuts and other cuts defined by case differentiations

After the somewhat natural presentation of the dual cuts in the last section, you already get the feeling that all cuts could be produced by dual deltas. Indeed we prove this in this section. Anyhow the proof is much more technical than the previous section.

For the inequality (set) y_j and the inequality y_k, both added separately to the same inequality system $\{y_m\}$, y_j is called harder than y_k when the solution space of the first system is contained in the second solution space.

In all of the proofs of cut generation schemas known to the author proofs by cases are used. So taking one of the cases of the proof, the set $\{y_j\}$ of that case should be harder than one arbitrary cut y_{result}, which will come from the method. If this is not the case, then it would be easy to define a new problem which will lead to a contradiction. Just leaving out all additional conditions and only adding the set $\{y_j\}$ will then already define the convex hull of the problem. So y_{result} will be not globally valid, which will be a contradiction. So the definition just made is sensitive and y_{result} can be considered as y_k.

For the following we always demand that for one inequality y_a the linear function $y_a(x)$ is the left-hand side of y_a. The scalar b_a should be the right-hand side. As always we are only considering greater or equal inequalities.

Now look at one LP consisting of the original inequalities $\{y_m\}$, the inequality (set) y_j and $y_k(x)$ as objective function. Via solving the problem to optimality we get a linear combination of $y_k(x)$ as dual solutions by the functions $\{y_m(x)\}$ and $y_j(x)$. Let us call this representation y^{rep}. As y_j is harder then y_k, the optimal point of this LP must have higher objective then b_k. So we get:

$$(\{b_m\}, b_j)y^{\text{rep}} - b_k > 0$$

$$y_k := (\{y_m\}, y_j)y^{\text{rep}} > (\{b_m\}, b_j)y^{\text{rep}} - ((\{b_m\}, b_j)y^{\text{rep}} - b_k)$$

From 7.1 we have some $\Delta^m(y_k^0)$ depending on the optimal dual solution y^0, which recreates y_k. We now use this representation to change the dual coordinates (y_k^0 is replaced by y_j^0). Therefor we define:

$$y_j^0 = (\{y_k^0\}_m + \{y^{\text{rep}}\}_m, (y^{\text{rep}})_j)$$

The aim is to show that this point is a dual solution and the branching cut defined by this point is similar but better compared to the cut y_j.

For the nth-dual equality in dual variables including $y_k : \sum_n a_{k,n}x_n \geq b_k$ we have:

$$c_n = \sum_m a_{m,n}(y_k^0)_m + a_{k,n}$$

Using the representation of y_k we get:

$$c_n = \sum_m a_{m,n}(y_k^0)_m + \sum_m a_{m,n}(y^{\text{rep}})_n + a_{j,n}(y^{\text{rep}})_j$$

$$c_n = \sum_m a_{m,n}(y_k^0 + y^{\text{rep}})_m + a_{j,n}(y^{\text{rep}})_j$$

So y_j^0 is also a solution. For the objective we have:

$$\omega(y_j^0) = \sum_m b_m((y_k^0)_m + (y^{\text{rep}})_m) + b_j(y^{\text{rep}})_j$$
$$= \omega(y_k^0) - b_k + (\{b_m\}, b_j)y^{\text{rep}} \geq \omega(y_k^0)$$

In the last calculation the fact that y_j is harder than y_k was used. For the Delta-value of the dual point y_j^0 we get:

$$\Delta^m(y_j^0) = \Delta^m(y_k^0) - (y^{\text{rep}})_m \leq \Delta^m(y_k^0)$$

We call the induced cut by the dual deltas of y_j^0 the inequality y_j^{cut}. So the following holds:

Remark 7.2 (Property of a harder inequality (set)) *For some inequality $y_k : \sum_m \Delta^m(y_k^0)s_m \geq \omega(y_k^0)$, the harder inequality y_j is possible to produce a cut y_j^{cut} with $\Delta^m(y_k^0) \geq \Delta^m(y_j^0)$ and $\omega(y_k^0) \leq \omega(y_j^0)$*

Doing the whole proof by cases via doing all maximizations and minimizations we get at once the final theorem:

Theorem 7.3 (Property of cuts defined by proof by cases) *An arbitrary cut y_k is produced by some cut method. Then the following form reproduces $y_k : \sum_m \Delta^m(y_k^0)s_m \geq \omega(y_k^0) - \omega(y')$ with $s_m = \sum_n a_{m,n}x_n - b_m$. Also there exists a branching cut inequality $\sum_m a_m s_m \geq b'$ with $a_m \leq \Delta^m(y_k^0)$ and $b' \geq \omega(y_k^0) - \omega(y')$. By definition this new cut is stronger than y_k.*

So the defined class dominates all known cuts for MILP.

It should be noted that for a cut with n normal case differentiations the above procedure uses 2^n different basic cases.

7.3 Conclusion

In the last two subsections we have built a powerful theory to study cut generations methods and their efficiency, which has some similarity to disjunctive cuts. Starting from one point y_0, before adding cuts or after having added some cuts, you try out all cuts in question in a row. For each cut y_i you got an representation by the $\Delta_m(y_i)$. For those representations you have the big combining inequality system from 2.9. By simplifications of it, you can do a lot of things with the cuts:

1. Get a quick view on where there are problems via the starting inequalities 2.7. This will give hints where collisions between cuts are.
2. Use the branch generation model from 6.1. We only proved it for one file (cut). Following these lines it can be shown that it is also applicable to generate more than one cut. In fact the model is dual just to the inequality, where all cuts have been added together, which is in fact the thing to be studied.

3. Using models where you try to optimize some inequalities, where weaker forms of them fit better together. So you will get better bounds. This can be reached via opening some of the inequalities subsystems of 2.9 at always the same variables.

Anyhow this study for each problem class and for each cut generation schema has to be done.
It might be possible that with this study there will be improvements in state of the art cut generators.

The author has implemented the cut generation method from 6.1. It worked perfectly and produced better values than the method 2.7. As this model is stronger than the model for the creation of the cuts this is expected. The author also did a few comparisons to other methods implemented in glpk. The textbook cut generation schemes were superior in quality than our general method. Anyhow these have been developed by quite a lot of people and for normal problems they are optimized by a lot experience. So it is no wonder that the effectiveness is not reached by a straightforward implementation of our method. Anyhow the idea of freezing almost all dual variables outside some neighbourhood of one certain variable from 5.3, was at this implementation already the most effective method, although the package glpk doesn't allow basis reuse in combination with problem presolve.

8 Exlicit description of the huge combining LP

This chapter is currently under construction....

This chapter may be considered as very technical. The author proposes readers first to understand all other sections, as the understanding the complexity of the huge combining LP might be difficult. This was also the case for the author of this work.

Before we have claimed, that all models are special forms of a very huge LP. In this section we really write down this LP, discuss all embedings, special cases and some consequences. Also the iteration of the dual combining LP will be explained. Here the limits will become understandable.

References

[1] V. Chvátal. Linear Programming, W.H. Freeman, New York, 1983. Discrete Mathematics Volume 4, Issue 4, April 1973, Pages 305-337

[2] M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. Mathematical Programming, 104(1):91104, 2005.

[3] V. Chvátal. Edmonds Polytopes and a hierarchy of combinatorical problems.

[4] Michael Perregaard. Generating Disjunctive Cuts for Mixed Integer Programs, Doctoral Dissertation, Carnegie Mellon University (2003)

[5] R.E. Bixby, S. Ceria, C.M. McZeal and M.W.P. Savelsbergh. An updated mixed integer programming library: MIPLIB 3.0, Optima 58 (1998) 12-15.

[6] M. Conforti, G. Cornuejols, G. Zambelli. Polyhedral Approaches to Mixed Integer Linear Programming. p. 343 - 385 in 50 Years of Integer Programming 1958–2008
Edited by M. Juneger at al.

[7] D. Applegate, R. E. Bixby, V. Chvátal, and W. Cook. Finding cuts in the TSP. Technical Report 95-05, DIMACS, March 1995