4 Column generation (CG)

- There are a lot of problems in integer programming where even the problem definition cannot be efficiently bounded
- Specifically, the number of columns becomes very large
- Therefore, these problems are hard to tackle by general algorithms
- Consider an optimal simplex tableau of such a problem:
  - Most variables are non-basic, so their value is 0
  - Only a tiny part of the matrix is of interest at all
- Basic question:

  How can we restrict unnecessary computational time that results from the consideration of unused variables?

4.1 Basics of column generation

- Basic Idea of column generation
  - Select a small subset of (promising) variables
  - Solve the corresponding LP-relaxation
  - Derive and solve a subproblem in order to identify whether there exists an unused variable, which would improve the objective value
  - If such a variable exists: include it and resolve the problem
  - If there is none: the problem is already solved to optimality

4.2 Applying CG to the Cutting-Stock Problem

- Certain materials (e.g., paper, metal) are manufactured in standard rolls of large width W
- This width is identical for all rolls
- These rolls are cut in smaller ones (called finals) $i=1,\ldots,m$ with widths w_i such that the number of sliced rolls is minimized
- Additionally, we have a demand of b_i finals of width w_i
- A solution defines in detail in which finals each roll is cut in order to
  - satisfy all demands and to
  - minimize the number of consumed rolls
- Clearly, a main problem arises by the fact that the entire solution space comprises a huge set of variables

Preparing the problem definition

- In what follows, we introduce the problem definition proposed by Gilmore and Gomory
- We define specific cutting patterns given by an integer vector $a=(a_1,\ldots,a_m)$
- It defines a specific feasible selection of a roll
- Altogether, we consider n feasible cutting patterns with

$$a^j=\left(a_1^j,\ldots,a_m^j\right),\text{with}\sum_{i=1}^{m}a_i^j\cdot w_i\leq W,\forall j\in\{1,\ldots,n\}$$

- Such a cutting pattern defines the segmentation of a roll into a set of finals of predetermined widths
- Consequently, we obtain the following linear integer program

The linear program and its dual

- We obtain the continuous primal problem *P*

$$\text{Minimize } Z = \sum_{j=1}^{n} x_j$$

$$\text{s.t. } \sum_{j=1}^{n} a_i^j \cdot x_j \geq b_i, \forall i \in \{1,\ldots,m\}$$

$$x_j \geq 0, \forall j \in \{1,\ldots,n\}, \text{with } \sum_{i=1}^{m} a_i^j \cdot w_i \leq W, \forall j \in \{1,\ldots,n\}$$

- and the corresponding continuous dual *D*

$$\text{Maximize } Y = \sum_{j=1}^{m} b_j \cdot y_j$$

$$\text{s.t. } \sum_{i=1}^{m} a_i^j \cdot y_i \leq 1, \forall j \in \{1,\ldots,n\}$$

$$y \geq 0$$

Intractability of the primal problem

- Clearly, the number of possible columns may become extremely large, even for small instances of the Cutting-Stock Problem
- Therefore, Gilmore and Gomory (1961) proposed column generation for solving its LP-relaxation
- First of all, an **initial solution of the LP-relaxation** is generated by defining a set of initializing columns
- We denote B as the matrix of the current columns

$$B^{ini} = \begin{pmatrix} d_1 = \left\lfloor \frac{W}{w_1} \right\rfloor & 0 & 0 & 0 \\ 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & d_m = \left\lfloor \frac{W}{w_m} \right\rfloor \end{pmatrix}$$

After solving the problem…

- We have a first row that tells us that the found solution in the simplex tableau is optimal
- Specifically, in our case, it is non-negative, i.e., it holds:

$$\bar{c}_i = c_i - \left(\pi^T \cdot A\right)_i \geq 0$$

- We substitute the parameters accordingly and obtain

$$\bar{c}_i = 1 - \left(y^T \cdot B\right)_i = 1 - \left(\sum_{j=1}^{m} y_j \cdot a_{j,i}\right) \geq 0 \Rightarrow \left(\sum_{j=1}^{m} y_j \cdot a_{j,i}\right) \leq 1$$

- Note that this applies only to the columns of matrix B
- Thus, there may be additional columns in A that lead to negative entries in the first row

Most attractive columns

- Thus, the larger $(a^i)\cdot y$ is, the more attractive becomes the column a^i to be integrated into the current tableau
- This directly results from the definition of the reduced costs in the primal tableau
- Therefore, in order to generate promising columns, we consider the following problem

$$\text{Maximize } Z = \sum_{j=1}^{m} a_j \cdot y_j$$

$$\text{s.t. } \sum_{j=1}^{m} a_j \cdot w_j \leq W$$

$$a_j \geq 0,\ a_j \text{ integer},\ \forall j \in \{1,\ldots,m\}$$

- Obviously, it is a **special Knapsack Problem**

Deriving a lower bound

- The optimal column determined by solving the special knapsack problem provides us with a lower bound of the optimal objective function value
- Clearly, for a current optimal dual solution y (optimal according to the current columns), it holds that

$$1-\underbrace{\left(y^T\cdot A\right)_{\max}}_{\substack{\text{Result of the best column}\\ \text{found by optimally solving}\\ \text{the special knapsack problem}}}\leq 1-\left(y^T\cdot A\right)_i,\forall i$$

$$\Rightarrow \left(y^T\cdot A\right)_{\max}\geq\left(y^T\cdot A\right)_i\Rightarrow\frac{\left(y^T\cdot A\right)_i}{\left(y^T\cdot A\right)_{\max}}\leq 1,\forall i$$

$$\Rightarrow\frac{y^T}{\left(y^T\cdot A\right)_{\max}}\cdot A\leq 1,\text{ we denote } z=\frac{y}{\left(y^T\cdot A\right)_{\max}}$$

- Apparently, z is a feasible dual solution to the LP-relaxation. Therefore, b^Tz provides us with a **lower bound**

Interpreting the lower bound

- This lower bound coincides with the objective value of the current optimal solution (i.e., the optimal solution corresponding to the active set of columns) with the objective function value

$$b^T \cdot y$$

- divided by the optimal value of the derived specifically designed knapsack subproblem for all possible columns

$$\left(y^T \cdot A\right)_{max}$$

Quality of the LP-relaxation bound

- The bound provided by the optimal solution of the LP-relaxation of Gilmore and Gomory's model is usually very tight (see Amor and Carvalho in Desaulniers, G.; Desrosiers, J.; Solomon, M.M.: Column Generation. p.137)
- "Specifically, most of the one dimensional cutting stock instances have gaps smaller than one, and we say that the instance has the integer round-up property, but there are instances with gaps equal to 1 (Marcotte, 1985, 1986), and as large as 7/6 (Rietz and Scheithauer, 2002). It has been conjectured that all instances have gaps smaller than 2, a property denoted as the modified integer round-up property (Scheithauer and Terno, 1995)"

Solution technique – Column generation

Hence, a pragmatic approach would be

- to solve the corresponding continuous problem by applying the Simplex Algorithm
- and round the resulting non-integer results accordingly
- Consequently, since we have m restrictions in the primal problem, we obtain an optimal solution with at most m non-zero variables
- Each rounding can cost us at most one additional roll
- Thus, the resulting cost difference between optimal continuous solution and found integer solution is upper bounded by m
- Note that this difference is extremely unlikely

General approach I

- We generate specific cutting patterns which are used as columns of a matrix B
- This matrix defines the following linear program

$$\text{Minimize } \sum_{i=1}^{m} x_i, \text{ s.t., } B \cdot x \geq b, x \geq 0 \qquad (*)$$

- If B is not directly available, we may use the diagonal matrix with the diagonal values d_i defined as follows

$$B^{ini} = \begin{pmatrix} d_1 = \left\lfloor \frac{W}{w_1} \right\rfloor & 0 & 0 & 0 \\ 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & d_m = \left\lfloor \frac{W}{w_m} \right\rfloor \end{pmatrix}$$

General approach II

- Then, we solve the defined problem optimally by applying the revised Simplex Algorithm
- This provides us with an optimal solution x^*
- However, its optimality according to the original matrix A depends on the definition of B
- Clearly, if the choice of cutting patterns was not appropriate, we may have generated a solution that will be outperformed by alternative constellations basing on modified columns
- But, in order to check this, we conduct the following steps

General approach III

- Subsequently, we calculate a dual optimal vector y with

$$y \cdot B = c_B \Leftrightarrow y = c_B \cdot B^{-1}$$

- Next, we try to find an integer vector $a=(a_1,\ldots,a_m)$, with $a_i \geq 0$ satisfying

$$\sum_{i=1}^{m} w_i \cdot a_i \leq W \text{ and } \sum_{i=1}^{m} y_i \cdot a_i > 1$$

- If such a vector exists, we replace one column in *B* by it (proposition)
- Otherwise, *x* defines an optimal solution to the continuous problem (proposition)

Proof of the proposition

- If we have no integer vector $a=(a_1,\ldots,a_m)^T$, with $a_i \geq 0$ satisfying

$$\sum_{i=1}^{m} w_i \cdot a_i \leq W \text{ and } \sum_{i=1}^{m} y_i \cdot a_i > 1$$

- we obviously know that for all

$$a = (a_1,\ldots,a_m) \text{ with } \sum_{i=1}^{m} w_i \cdot a_i \leq W, \text{ it holds } \sum_{i=1}^{m} y_i \cdot a_i \leq 1$$

- Thus, we may conclude that y is a feasible dual solution and the complete simplex tableau has a positive first row

Proof of the proposition

- This first row is defined as

$$\bar{c}_i = c_i - \left(\pi^T \cdot A\right)_i = 1 - \underbrace{\left(\sum_{j=1}^{m} y_j \cdot a_{j,i}\right)}_{\leq 1} \geq 0$$

  and therefore the found solution is optimal
- We cannot improve the found optimal solution of the reduced problem (*) by integrating any additional cutting pattern, i.e., by integrating any additional column
- Additionally, the objective value of *x** provides a lower bound on the objective function value of the best integer solution

Conclusion

- We apply the revised Simplex Algorithm to a linear program
- Then, it is not necessary that the whole matrix A_N of non-basic columns is available
- Moreover, it is sufficient to store the current base matrix B and to have a procedure at hand which calculates an entering column a_j (i.e., a column a_j of A_N satisfying $y \cdot a_j > c_j$ (MinProb)), or proves that no such column exists
- This **problem is denoted as the pricing problem** and is solved by a **pricing procedure**

Pricing procedure

- Usually, a pricing procedure does not calculate only a single column but a set of columns which possibly may enter the basis in the following iterations
- Thus, we have always a so-called **working set of active columns**
- If, finally, the pricing procedure states that no entering column exists, the current basic solution is optimal and the algorithm terminates

Column generation algorithm

1. Initialize
2. WHILE Calculate Columns produces new columns DO
   - Insert and Delete Columns
   - Optimize
3. END

➢ Note that only Initialize and Calculate Columns have to be implemented problem-specifically

Example

- We introduce the following simple example
  - We have rolls of size W=100 and need
    - 97 finals of width 45
    - 610 finals of width 36
    - 395 finals of width 31
    - 211 finals of width 14
  - Additionally, we may apply the following cutting patterns

$$a^1 = \begin{pmatrix} 2 \\ 0 \\ 0 \\ 0 \end{pmatrix}, a^2 = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \end{pmatrix}, a^3 = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \end{pmatrix}, \text{and } a^4 = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 0 \end{pmatrix}$$

Example

- Thus, we get the system

$$B \cdot x \geq \begin{pmatrix} 97 \\ 610 \\ 395 \\ 211 \end{pmatrix} \Leftrightarrow \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 2 & 1 \\ 0 & 0 & 0 & 2 \\ 0 & 2 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \geq \begin{pmatrix} 97 \\ 610 \\ 395 \\ 211 \end{pmatrix}$$

- This system has the optimal continuous solution

$$x = \begin{pmatrix} 48{,}5 \\ 105{,}5 \\ 100{,}75 \\ 197{,}5 \end{pmatrix} \wedge x_j = 0, \forall j = 5, \ldots, n \quad \text{and the dual solution } y = \begin{pmatrix} 0{,}5 \\ 0{,}5 \\ 0{,}25 \\ 0 \end{pmatrix}.$$

Dual solution y

- We consider all dual solutions *y*
- It is defined by

$$y \cdot B \leq c \Leftrightarrow \begin{pmatrix} y_1 & y_2 & y_3 & y_4 \end{pmatrix} \cdot \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 2 & 1 \\ 0 & 0 & 0 & 2 \\ 0 & 2 & 0 & 0 \end{pmatrix} \leq \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \Leftrightarrow \left| \begin{array}{c} 2 \cdot y_1 \leq 1 \\ 2 \cdot (y_2 + y_4) \leq 1 \\ 2 \cdot y_2 \leq 1 \\ y_2 + 2 \cdot y_3 \leq 1 \end{array} \right|$$

Thus, we obtain…

- for the corresponding optimal dual solution

$$c_B \cdot A_B^{-1} = y^T = \begin{pmatrix} 0{,}5 & 0{,}5 & 0{,}25 & 0 \end{pmatrix}$$

- Let us now consider

$$y_1 \cdot a_1 + \ldots + y_4 \cdot a_4 = 0{,}5 \cdot a_1 + 0{,}5 \cdot a_2 + 0{,}25 \cdot a_3$$

- We have to show that it holds:

$$y_1 \cdot a_1 + \ldots + y_4 \cdot a_4 = 0{,}5 \cdot a_1 + 0{,}5 \cdot a_2 + 0{,}25 \cdot a_3 \leq 1$$

- Thus, we obtain

$$50 \cdot a_1 + 50 \cdot a_2 + 25 \cdot a_3 \leq 100$$

- In what follows, we analyze feasible cutting patterns for *a*

Restrictions

- A feasible cutting pattern *a* must fulfill the following restrictions
- First, it must not consume more widths than the roll contains, i.e.,

$$45 \cdot a_1 + 36 \cdot a_2 + 31 \cdot a_3 + 14 \cdot a_4 \leq 100 \quad (1)$$

- In addition, our found subset of cutting patterns is optimal if it holds for all cutting patterns that

$$50 \cdot a_1 + 50 \cdot a_2 + 25 \cdot a_3 \leq 100$$

- Thus, we assume to the contrary that it holds:

$$50 \cdot a_1 + 50 \cdot a_2 + 25 \cdot a_3 > 100 \quad (2)$$

Consequences

- If a exists, then we know at first that $a_3=0$
- Why?
  - If $a_3>3$, we have a direct contradiction to (1)
  - If $a_3=3$, we have $a_1=a_2=0$ due to (1), but this contradicts (2) since 75>100 is obviously not correct
  - If $a_3=2$, we have $a_1=0$ and $a_2\leq 1$ due to (1), but this contradicts (2) since 100>100 is obviously not correct
  - If $a_3=1$, we have $a_1+a_2\leq 1$ due to (1), but this contradicts (2) since 75>100 is obviously not correct
  - Consequently, we obtain $a_3=0$ as claimed

Consequences

- Thus, we have a_3=0 and thus we may write now a modified system

$$45 \cdot a_1 + 36 \cdot a_2 + 14 \cdot a_4 \leq 100 \quad (1)$$

- and

$$50 \cdot a_1 + 50 \cdot a_2 > 100 \quad (2)$$

- Consequently, since (1) we may conclude that $a_1+a_2 \leq 2$, and therefore we again have a contradiction to (2)
- Consequently, a does not exist at all
- Thus, x^{*T}=(48,5;105,5;100,75; 197.5) is an optimal solution for the continuous relaxation of our problem P and has the objective function value 452,25

An optimal integer solution

- We transform the continuous solution x^{*T}=(48,5; 105,5; 100,75; 197,5) to x^{1^T}=(48, 105, 100, 197)
- Thus, we apply the following constellation
  - 48 times cutting pattern (2,0,0,0),
  - 105 times cutting pattern (0,2,0,2),
  - 100 times cutting pattern (0,2,0,0), and
  - 197 times cutting pattern (0,1,2,0)
- Result
  - 96 finals of width 45 (Demand 97)
  - 607 finals of width 36 (Demand 610)
  - 394 finals of width 31 (Demand 395)
  - 210 finals of width 14 (Demand 211)

An optimal integer solution

- Hence, we have altogether 450 rolls of width *W*
- This, however, is an infeasible constellation, but we may add three additional patterns of the forms (0,2,0,0), (1,0,1,1), and (0,1,0,0)
- Consequently, we obtain a feasible solution that consumes altogether 453 rolls
- Since the optimal continuous solution has an objective function value of 452,25, the integer solution x^1 is optimal

Finding integer solution in general…

- Column generation is a technique that enables us to efficiently solve problems with a large number of columns
- If we have optimally solved the corresponding continuous problem without finding an integer solution but a fractional one, we may
  - branch by fix a non-integer variable and repeat the solution process (Branch&Price)
  - round variables accordingly
  - apply a specifically designed metaheuristic
  - combine some of the methods depicted above in a sophisticated way
- Clearly, it depends on the application how useful a found optimal solution of the LP-relaxation is
- In case of the Cutting-Stock Problem, these solutions directly provide us with tight bounds
  - Basically, the subproblem is the Knapsack Problem that, despite its NP-Completeness, can be solved quite efficiently
  - Therefore, in this case, the subproblem step works quite smoothly

References of Section 4

- Amor, H.B.; Valério de Carvalho, J: Cutting Stock Problems. In Desaulniers, G.; Desrosiers, J.; Solomon, M.M. (eds.): *Column Generation*. Springer, pp. 131-162, 2005.
- Danna, E.; Pape, C.L.: Branch-and-Price Heuristics: A Case Study on the Vehicle Routing Problem with Time Windows. In Desaulniers, G.; Desrosiers, J.; Solomon, M.M. (eds.): *Column Generation*. Springer, pp. 99-130, 2005.
- Desrosiers, J.; Lübbecke, M.E.: A Primer in Column Generation. In Desaulniers, G.; Desrosiers, J.; Solomon, M.M. (eds.): *Column Generation*. Springer, pp. 1-32, 2005.
- Gilmore, P. and Gomory, R.: A linear programming approach to the cutting stock problem. *Operations Research* 9:849-859, 1961.

References of Section 4

- Gilmore, P. and Gomory, R.: A linear programming approach to the cutting stock problem-Part 2. *Operations Research* 11: 863-888, 1963.
- Gouhmis, C.: Optimal solutions to the cutting stock problem. *European Journal of Operational Research* 44:197-208, 1990.
- Kallehauge, B.; Larsen, J.; Madsen, B.G.; Solomon, M.M.: Vehicle Routing Problem with Time Windows. In Desaulniers, G.; Desrosiers, J.; Solomon, M.M. (eds.): *Column Generation*. Springer, pp. 67-98, 2005.
- Rietz, J. and Scheithauer, G.: Tighter bounds for the gap and non-IRUP constructions in the one-dimensional cutting stock problem. *Optimization*, 51(6):927-963, 2002.

References of Section 4

- Scheithauer, G. and Terno, J.: The modified integer roundup property of the one-dimensional cutting stock problem. *European Journal of Operational Research*, 84:562-571, 1995.