Chapter 9

A column generation/simplicial decomposition algorithm

9.1 Column generation approaches

9.1.1 Background

The solution strategy known as the column generation principle—and its dual counterpart, constraint generation—is one of the standard tools of mathematical programming. Since the pioneering work on the maximum multi-commodity network flow problem by Ford and Fulkerson [FoF58, FoF62] in the late 50's, column generation methods have been developed and applied in a variety of contexts. With no doubt, the most famous column generation method is the Dantzig–Wolfe decomposition method for specially structured linear programs ([DaW60, DaW61]), in which the column generation is based on the pricing-out mechanism of the simplex method. [Another famous column generation technique is the one for cutting stock problems, where the column generation is based on the solution of knapsack problems ([GiG61, GiG63, GiG65]).]

Simplicial decomposition (SD) is a class of methods for solving continuous problems in mathematical programming with convex feasible sets. There are two main characteristics of the methods in this class: (1) an approximation of the original problem is constructed and solved, wherein the original feasible set is replaced by a polyhedral subset thereof, that is, an inner approximation of it which is spanned by a finite set of feasible solutions; and (2) this inner approximation is improved (that is, enlarged) by generating a vector (or, column) in the feasible set through the solution of another approximation of the original problem wherein the original cost function is approximated (often by a linear function). As such, the class of SD methods may be placed within the framework of column generation methods. Another characteristic of an SD method
is that the sequence of solutions to the inner approximated problems tends to a solution to the original problem in such a way that some merit function strictly monotonically approaches its optimal value. Therefore, the class of SD methods also falls within the framework of iterative descent algorithms for continuous mathematical programs, and this observation is important for the development of the new algorithm to follow.

## 9.1.2 Inner representation

The derivation of the SD methods rests on two classic results on the representation of convex sets and of points in such sets. The first result is the Representation Theorem (e.g., [Las70, Thm. 3.2] or [BSS93, Thm. 2.6.7]), which states that (1) the set of extreme points $p^i$, $i \in P$, of a nonempty polyhedral set $X$ is nonempty and finite; (2) the set of extreme directions $d^i$, $i \in D$, is empty if and only if $X$ is bounded, and if $X$ is not bounded, then it is (nonempty and) finite; finally, and most importantly, (3) a vector $x \in \mathbb{R}^n$ belongs to $X$ if and only if it can be represented as a convex combination of the extreme points plus a non-negative linear combination of the extreme directions, that is, for some vectors $\lambda$ and $\mu$,

\[
\begin{align*}
x &= \sum_{i \in P} \lambda_i p^i + \sum_{i \in D} \mu_i d^i, \\
\sum_{i \in P} \lambda_i &= 1, \\
\lambda_i &\geq 0, \quad i \in P, \\
\mu_i &\geq 0, \quad i \in D.
\end{align*}
\]

We illustrate the notions of outer and inner representations through the traffic equilibrium problem, stated in Example 1.9 and further discussed in Example 8.2.

**Example 9.1 (Traffic equilibrium).** We introduce some further notation. We let $v_{ijk}$ denote the flow on link $(i,j) \in A$ in commodity $k$. A feasible flow for commodity $k$ then is a vector $v_k$ satisfying

\[
\begin{align*}
\sum_{j \in W_i} v_{ijk} - \sum_{j \in V_i} v_{jik} &= d_{ik}, \quad i \in N, \\
v_{ijk} &\geq 0, \quad (i,j) \in A,
\end{align*}
\]

where

\[
d_{ik} = \begin{cases} 
d(k), & \text{if node } i \text{ is the origin of commodity } k, \\
-d(k), & \text{if node } i \text{ is the destination of commodity } k, \\
0, & \text{otherwise,}
\end{cases} \quad i \in N,
\]

and

\[
W_i = \{ j \mid (i,j) \in A \}; \quad V_i = \{ j \mid (j,i) \in A \}
\]