Dr. Niels Lindner (lindner@zib.de)
Pedro Maristany de las Casas (maristany@zib.de)

Lecture 15

February 3, 2020

6.3 Multi-Depot Aperiodic Vehicle Scheduling

6.3.2 Path-based multi-commodity flow

For the multi-depot vehicle scheduling problem, any feasible multi-commodity flow can be transformed into a set of certain p_d-q_d-paths. Let $\mathcal{P}_d$ denote the set of all p_d-q_d-paths for a depot $d \in \mathcal{D}$.

Corollary 1. *The multi-depot vehicle scheduling problem is solved by finding subsets $P_d \subseteq \mathcal{P}_d$ for each $d \in \mathcal{D}$ such that*

(1) *For each trip $t \in \mathcal{T}$ there is a feasible depot $d \in D(t)$ such that $(d_t, a_t) \in E(p)$ for some $p \in P_d$, and*

(2) *$\sum_{d \in \mathcal{D}} |P_d|$ is minimum.*

For an edge $e = (d_t, a_t) \in E_d$, define $\mathcal{P}_e := \bigcup_{d \in D(t)} \{p \in \mathcal{P}_d \mid e \in E(p)\}$. Then the multi-depot vehicle scheduling problem is solved by

$$\begin{aligned} \text{Minimize} \quad & \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d} f_p & \\ \text{s.t.} \quad & \sum_{p \in \mathcal{P}_e} f_p = 1, & e \in E_d, \\ & f_p \in \{0, 1\}, & p \in \mathcal{P}_d, d \in \mathcal{D}. \end{aligned}$$

This is the *path-based multi-commodity flow formulation* of the multi-depot vehicle scheduling problem, sometimes also called *set partitioning formulation*. Note however that the number of paths is enormous (typically exponential), so that it is hard – or simply impossible – to write down all variables and constraints explicitly. On the other hand, almost all variables f_p will be set to 0 in any feasible solution: Since the trivial schedule uses $|E_d|$ paths, and we minimize the number of total paths, an optimal solution will have at most $|E_d|$ non-zero variables.

Solving the LP relaxation, e.g., in the context of branch-and-bound, can be achieved by techniques such as *column generation* (according to a 2012 survey of Nemhauser, invented by Ford and Fulkerson, 1971).

Column Generation

Starting with variables restricted to any feasible solution, e.g., the trivial schedule, where vehicles immediately pull in after their first trip, one determines which variable (*column*) to add to the problem in order to decrease the objective value (*pricing*). In other words, we solve first the LP relaxation

$$\begin{array}{lll} \text{Minimize} & \displaystyle\sum_{p \in \mathcal{P}'} f_p & \\ \text{s.t.} & \displaystyle\sum_{p \in \mathcal{P}_e \cap \mathcal{P}'} f_p = 1, & e \in E_d, \\ & f_p \geq 0, & p \in \mathcal{P}'. \end{array}$$

for a subset $\mathcal{P}' \subseteq \mathcal{P} := \bigcup_{d \in \mathcal{D}} \mathcal{P}_d$ of paths. The optimal solution f^* corresponds to an optimal solution x^* to the dual LP

$$\begin{array}{lll} \text{Maximize} & \displaystyle\sum_{e \in E_d} x_e & \\ \text{s.t.} & \displaystyle\sum_{e \in E_d \cap E(p)} x_e \leq 1, & p \in \mathcal{P}'. \end{array}$$

Although f^* is always feasible for the unrestricted primal LP, x^* might not be feasible for the unrestricted dual LP, i.e., there is a path $p \in \mathcal{P} \setminus \mathcal{P}'$ such that $\sum_{e \in E_d \cap E(p)} x_e^* > 1$. If x^* happens to be feasible for the unrestricted dual LP, then, by LP duality, f^* must have been optimal for the unrestricted primal LP, and vice versa. Hence

$$\begin{aligned} & f^* \text{ is optimal for the unrestricted primal LP} \\ & \Leftrightarrow x^* \text{ is feasible for the unrestricted primal LP} \\ & \Leftrightarrow \forall p \in \mathcal{P} : \sum_{e \in E_d \cap E(p)} x_e^* \leq 1 \\ & \Leftrightarrow \max_{p \in \mathcal{P}} \sum_{e \in E_d \cap E(p)} x_e^* \leq 1. \end{aligned}$$

The *pricing problem* is hence to find a longest path $p^* \in \mathcal{P}$ w.r.t. the costs given by x^* on the driving activities and 0 on all other activities. If p^* has cost ≤ 1, then f^* has been optimal for the unrestricted LP. Otherwise, we can add the variable f_{p^*} to the primal LP and the constraint for p^* to the dual LP, and repeat.

Since the network is acyclic, this longest path problem w.r.t. x^* can be found in polynomial time: Simply look for a shortest path with costs $-x^*$. There are no negative cost cycles, as there are no directed cycles at all. In fact, there is even a linear time algorithm based on *topological search*. We will discuss topological search in detail as a shortest path algorithm for public transportation networks next semester, and column generation for multi-commodity flow will return in the context of line planning.

Extensions

Our multi-depot aperiodic vehicle scheduling model can easily be extended to cope with the following aspects:

- *Depot capacities*: A maximum number κ_d of vehicles that can be parked/maintained at a depot $d \in \mathcal{D}$ can be modeled by requiring
$$\sum_{p \in \mathcal{P}_d} f_p \leq \kappa_d.$$
- *Multiple vehicle types*: Modeled by "virtual" depots for each feasible pair of vehicle type and depot.
- *Operational costs*: Path-depending costs c_p (e.g., fuel) can be dealt with by minimizing $\sum_p c_p f_p$. Note that this makes the pricing problem hard unless the costs are aggregated from the edges.
- *Route constraints*: Remove paths violating, e.g., length constraints. In this case, the pricing problem becomes now a *resource-constrained shortest path problem*, which is NP-hard even on directed acyclic graphs.
- *Electric vehicle scheduling*: In principle, the path-based multi-commodity flow version is general enough to deal with the special needs of electric vehicles. Recharging batteries of electric vehicles can either be done in a depot, or on dedicated recharging stations typically located at endpoints of lines. Both versions can be modeled by restricting the set of feasible paths $\mathcal{P}$. A simple way is to bound the lengths of the paths conforming to the vehicle ranges. However, more detailed models price or pre-compute the paths taking into account battery capacities, current state of charge, energy consumption, as well as time constraints and overcrowding at recharging stations.
- *Duty scheduling*: Duties of drivers, pilots, conductors etc. are sequences of *duty elements.* E.g., a duty for a bus driver could be to drive a bus from A to B, and later another one from B to C. Of course, several constraints have to be met concerning, e.g., breaks, hours of work, night and weekend duties and so on. However, in principle, scheduling duties for drivers means to find certain feasible sequences of (partial) trips, so that the problem is very similar to vehicle scheduling. In fact, both problems can be treated in an *integrated* way, i.e., vehicle and duty paths can be computed within the same optimization problem, and even on the same acyclic network (see e.g., Borndörfer, Löbel, Weider, 2008). However, regulations for duties produce more sophisticated pricing problems.
- *Crew scheduling*: The task of crew scheduling is to assign real people to duties. Again, this can be modeled as a multi-commodity flow or set partitioning problem similar to vehicle scheduling, taking into account the special requirements of the employees: For example, drivers should not be scheduled too far away from their homebases, there must be sufficient leisure time between two duties etc.

6.4 Vehicle Routing

A generalization of both ATSP and Single-Depot Vehicle Scheduling is the *Vehicle Routing Problem.* The problem dates back to Dantzig and Ramser (1959). Starting from a depot, a fleet of trucks needs to deliver goods to customers, taking into account truck capacities and the total distance traveled. Mathematically, the problem is formulated as follows:

Definition 2. *Given*

- *a complete digraph K_n^* on the vertices $V(K_n^*) = \{0, 1, \dots, n-1\}$,*
- *a cost function $c : E(K_n^*) \to \mathbb{R}_{\geq 0}$,*
- *a demand function $q : V(K_n^*) \to \mathbb{R}_{\geq 0}$ with $q(0) = 0$,*
- *a vehicle capacity $Q \geq 0$,*
- *a fleet size $K \in \mathbb{N}$,*

the Capacitated Vehicle Routing Problem (CVRP) *is to find a set $\{C_1, \dots, C_K\}$ of directed circuits such that*

(1) $0 \in C_i$, $i = 1, \dots, K$,

(2) $\sum_{v \in V(C_i)} q_v \leq Q$, , $i = 1, \dots, K$,

(3) $V(C_i) \cap V(C_j) = \{0\}$, $i \neq j$,

(4) $\bigcup_{i=1}^K V(C_i) = V(K_n^*)$,

(5) $\sum_{i=1}^K \sum_{e \in E(C_i)} c(e)$ *is minimum.*

Example 3. For $K = 1$ and $q \equiv 0$, this is nothing but ATSP. In Exercise 2 on Problem Set 13, the connection with single-periodic vehicle scheduling has been outlined.

Integer Programming Formulations

The following is an integer programming formulation of CVRP:

$$\begin{array}{lll}
\text{Minimize} & \displaystyle\sum_{e \in E(K_n^*)} c_e x_e & \\
\text{s.t.} & \displaystyle\sum_{e \in \delta^+(v)} x_e = 1, & v \in V(K_n^*) \setminus \{0\}, \\
& \displaystyle\sum_{e \in \delta^-(v)} x_e = 1, & v \in V(K_n^*) \setminus \{0\}, \\
& \displaystyle\sum_{e \in \delta^+(0)} x_e = K, & \\
& \displaystyle\sum_{e \in \delta^+(S)} x_e \geq r(S), & \emptyset \neq S \subseteq V(K_n^*) \setminus \{0\}, \\
& x_e \in \{0, 1\}, & e \in E(K_n^*).
\end{array}$$

Here, $r(S)$ denotes the number vehicles necessary for covering the demand at S. A precise value for $r(S)$ can be obtained by solving the *bin packing* problem with bins of capacity Q and item weights q. Observe that $r(S) \geq \lceil \sum_{v \in S} q(v)/Q \rceil \geq 1$. In particular, every feasible solution x of the above integer program satisfies the subtour elimination constraints

$$\sum_{e \in \delta^+(S)} x_e \geq 1, \quad \emptyset \neq S \subseteq V(K_n^*) \setminus \{0\}.$$

On the computational side, many TSP heuristics carry over to vehicle routing, and various specific cuts are known, see, e.g., the survey book by Toth and Vigo (2002).

From the path-based or set partitioning perspective, let $\mathcal{C}$ be the set of *feasible routes*, i.e., the set of directed circuits C such that $0 \in V(C)$ and $\sum_{v \in V} q_v \leq Q$. Then CVRP can be written as the following path-based integer program:

$$\begin{array}{lll}
\text{Minimize} & \displaystyle\sum_{C \in \mathcal{C}} \sum_{e \in E(C)} c_e y_C & \\
\text{s.t.} & \displaystyle\sum_{C \in \mathcal{C} \colon v \in V(C)} y_C \geq 1, & v \in V(K_n^*) \setminus \{0\}, \\
, & \displaystyle\sum_{C \in \mathcal{C}} y_C = K, & \\
& y_C \in \{0, 1\}, & C \in \mathcal{C}.
\end{array}$$

Again, the size of $\mathcal{C}$ is exponential, and solving this IP will require column generation.

Dial-a-Ride, or Ride Pooling

The *dial-a-ride problem (DARP)* combines pickup and delivery of passengers. Examples are US-style school buses, transport services for disabled people, or ride pooling such as BerlKönig or CleverShuttle. In addition to the requirements of capacitated vehicle routing, the following has to be respected in a DARP (Doerner and Salazar-González, 2002):

(1) Pickup-Delivery-Coupling: Pickup and delivery of a customer must be accomplished by the same vehicle.

(2) Precedence: Each customer must be picked up before being delivered.

(3) Time windows: Pickup and delivery should lie within given time intervals.

(4) Maximum travel time: The time a costumer spends in a vehicle should not be too large.

If all these constraints are considered hard, i.e., mandatory for any feasible solution, then DARP can be modeled as a CVRP with the additional constraints

$$\sum_{e \in R} x_e \leq |R| - 1, \quad R \in \mathcal{R},$$

Dr. Niels Lindner (lindner@zib.de)
Pedro Maristany de las Casas (maristany@zib.de)

where $\mathcal{R}$ is a set containing (sub)routes R violating one of the above conditions. For example, the coupling constraint can be modeled as

$$\sum_{e \in E[S]} x_e \leq |S| - 1$$

for all subsets $S \subseteq V(K_n^*)$ containing 0 and some pick-up vertex v, but not the corresponding delivery vertex for v. Unsurprisingly, these constraints cannot be written down explicitly, but they can be used in a cutting plane approach: The coupling constraints can be separated by minimum cuts just as the subtour elimination constraints for TSP. To check whether a given route satisfies time window and maximum travel time constraints, one can solve a shortest path problem on a vertex-weighted interval graph (Firat and Woeginger, 2011).