CE 205A
Transportation Logistics

Lecture 19

Vehicle Scheduling Problem

Lecture Outline

1. Single Depot VSP
2. Multi Depot VSP

Single Depot VSP

Single Depot VSP

Introduction

The Vehicle Scheduling Problem (VSP) deals with assigning vehicles (buses, flights, ferries, trains) to scheduled time tables.

Assume that we are given a set of trips $\{T_1, \ldots, T_n\}$ and a single depot D.

Let each trip i have a departure/start time d_i and arrival/end time a_i.

Also suppose that trip i starts at a location p_i and terminates at a location q_i.

Single Depot VSP

Introduction

We assume that all vehicles start at the depot and return back to it.

Vehicles can deadhead from q_i to p_j without carrying passengers and incur a cost c_{ij} and take t_{ij} units of time.

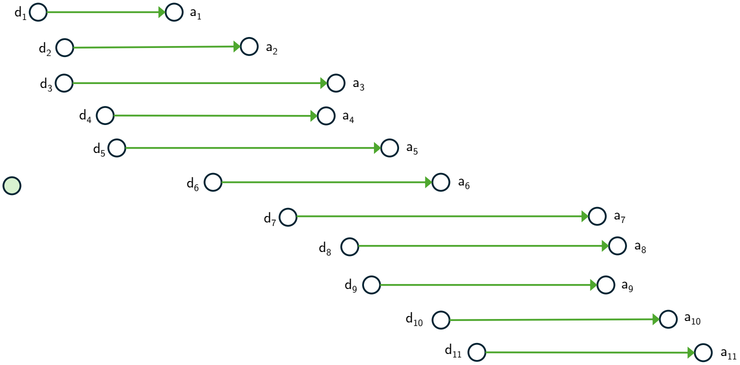
Deadheading between trips T_i and T_j are allowed only if they are *compatible*, i.e., $a_i + t_{ij} > d_j$.

The goal is usually either minimize the number of vehicles used or the total cost of deadheading.

Single Depot VSP

Network Flow Formulation – Version I

There are multiple ways to formulate this problem. We first create a graph $G = (N, A)$ with departure event nodes, arrival event nodes, and the depot (or two copies of it say D_d and D_a).

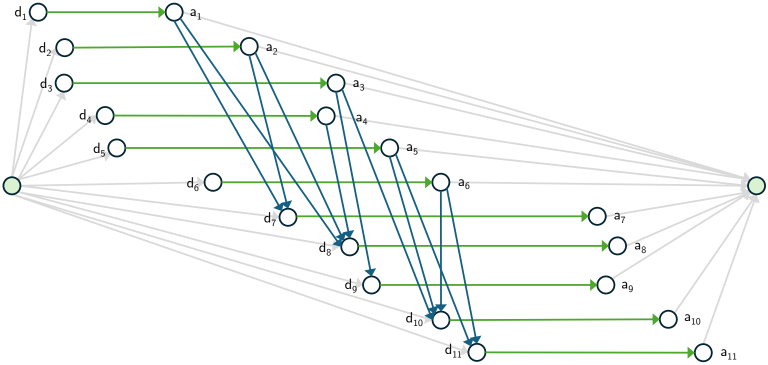


Single Depot VSP

Network Flow Formulation – Version I

Three types of edges are included in the graph:

- Pull-in and pull-out arcs from and to depot (grey) A_1
- Trip arcs (green) A_2
- Compatible arcs (blue) A_3



Single Depot VSP

Network Flow Formulation – Version I

Only deadheading of pull-in and pull-out arcs and compatible arcs matter. (Why?)

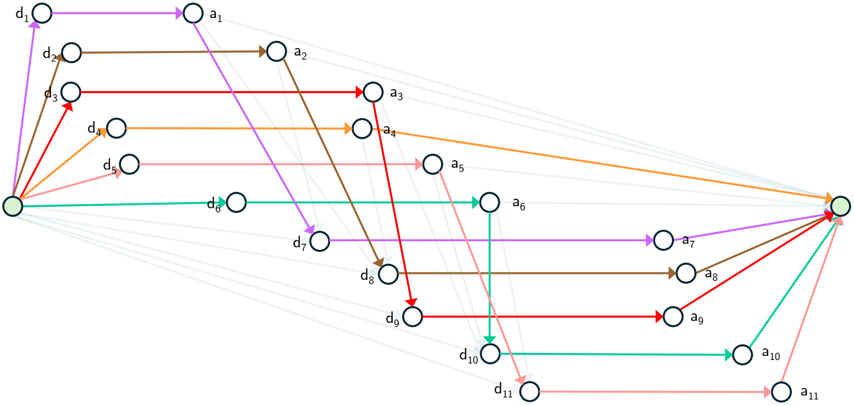
Formulate the problem as a min-cost network flow problem.

$$
\begin{array}{lll}
\min & \displaystyle\sum_{(i,j)\in A_1\cup A_3} c_{ij}x_{ij} & \\
\text{s.t.} & x(\delta^+(i)) = x(\delta^-(i)) & i \in \{d_1,\ldots,d_n\}\cup\{a_1,\ldots,a_n\} \\
& x_{ij} = 1 & \forall\,(i,j)\in A_2 \\
& x_{ij}\in\{0,1\} & \forall\,(i,j)\in A
\end{array}
$$

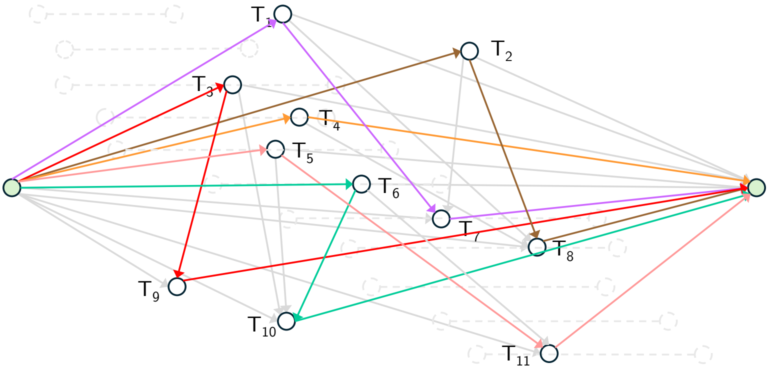
Is constraint matrix totally unimodular? How do we model the problem of minimizing the number of vehicles used?

Single Depot VSP

Solution



Alternatively instead of representing events, a trip is represented as a node. Write a network flow formulation for this network.



(Trips have been moved horizontally for visualization purposes.)

Single Depot VSP

Network Flow Formulation – Version II

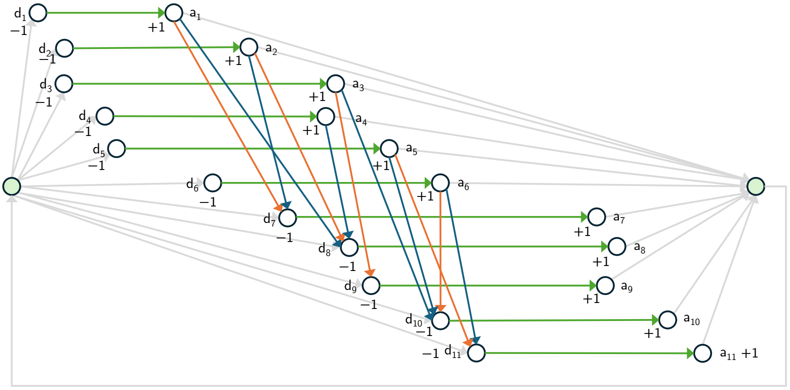
$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} & \\
\text{s.t.} \quad & x(\delta^+(i)) = 1 & i \in \{T_1,\ldots,T_n\} \\
& x(\delta^+(i)) = x(\delta^-(i)) & \forall\, i \in \{T_1,\ldots,T_n\} \\
& x_{ij} \in \{0,1\} & \forall\,(i,j) \in A
\end{aligned}
$$

One could also impose restrictions on the fleet sizes using additional constraints.

Single Depot VSP

Network Flow Formulation – Version III

A third formulation can be written that moves the trip flows to demands to make it appear in a more familiar form on a slightly modified network.



Single Depot VSP

Network Flow Formulation – Version III

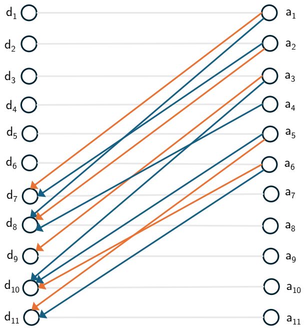
Let b_i indicate the demands which is set to $+1$ for the arrival nodes and -1 for the departure nodes.

$$\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij}x_{ij} \\
\text{s.t.} \quad & x(\delta^{+}(i)) - x(\delta^{-}(i)) = b_i \,\forall\, i \in N \\
& 0 \leq x_{ij} \leq 1 \,\forall\, (i,j) \in A
\end{aligned}$$

Single Depot VSP

Matching Problem

The SDVSP problem can also be modeled as a bipartite matching problem using the compatible edges when minimizing the number of vehicles.



The optimal solution is $n-$ optimal cardinality of the above matching. (Why?)

Multi Depot VSP

Multi Depot VSP

Introduction

The SDVSP can be solved in polynomial time. However, in most scenarios there are multiple depots.

Suppose the system has depots $D_1, D_2, \ldots, D_k$. Each depot is assumed to have a capacity ν_k. Let $K = \{1, 2, \ldots, k\}$

Suppose vehicles can start at any depot and end at any depot as long as the capacity limits are met.

Can you solve this using an optimization model? What is the complexity of this variant.

Multi Depot VSP

Introduction

In the more general setting, vehicles are tagged to depots and start and end their trips at the same depot.

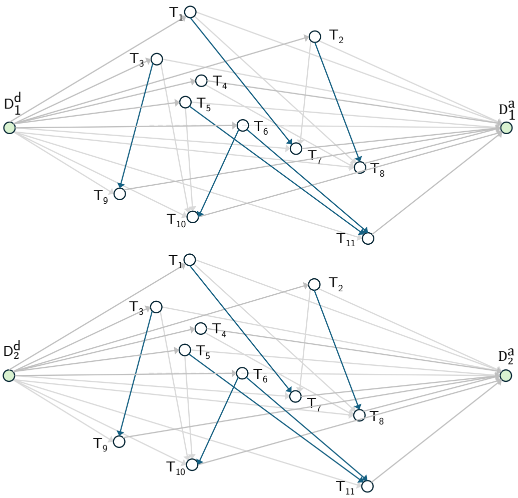
This helps with servicing and maintenance activities. Formulate this version using an optimization model.

Create k layers $G^k = (V^k, A^k)$ or copies of the network, where $V^k = \{T_1, \ldots, T_n, D^a_k, D^d_k\}$. The arc set for each graph is similar to before.

The pull-in and pull-out arc costs can now be depot-dependent.

Multi Depot VSP

Introduction



Multi Depot VSP

Introduction

Let x_{ij}^k be 1 if a link (i, j) in layer k is traversed and is 0 otherwise.

$$
\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{(i,j) \in A^k} c_{ij}^k x_{ij}^k && \\
\text{s.t.} \quad & \sum_{k \in K} x^k(\delta^+(i)) = 1 && i \in \{T_1, \ldots, T_n\} \\
& x^k(\delta^+(i)) = x^k(\delta^-(i)) && \forall\, i \in \{T_1, \ldots, T_n\}, k \in K \\
& x^k(\delta^+(D_k^d)) \leq \nu_k && \forall k \in K \\
& x_{ij}^k \in \{0, 1\} && \forall\, (i, j) \in A^k, k \in K
\end{aligned}
$$

Multi Depot VSP

Column Generation

Just as the VRP, one can also formulate this using a set-packing formulation. This allows us to impose rotation/duty-specific constraints.

Let Ω_k be rotations that start from a depot D_k, visits a few trips and comes back to the same depot.

$$
\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{p \in \Omega_k} c_p y_p \\
\text{s.t.} \quad & \sum_{p \in \Omega_k} y_p \leq \nu_k && \forall\, k \in K \\
& \sum_{k \in K} \sum_{p \in \Omega_k} a_{ip} y_p = 1 && \forall\, i \in \{T_1, \ldots, T_n\} \\
& y_p \in \{0, 1\} && \forall\, p \in \Omega_k, k \in K
\end{aligned}
$$

Multi Depot VSP

Column Generation

Can you write the pricing problem formulation for the LP relaxation of the RMP?

$$
\begin{aligned}
\min \quad & -\sigma_k + \sum_{(i,j)\in A^k} (c_{ij} - \pi_j) x_{ij}^k \\
\text{s.t.} \quad & x^k(\delta^+(i)) - x^k(\delta^-(i)) = \begin{cases} 1 & \text{if } i = D_k^d \\ -1 & \text{if } i = D_k^a \\ 0 & \text{otherwise} \end{cases} \\
& x_{ij}^k \in \geq 0 \qquad \forall\, k \in K, (i,j) \in A^k
\end{aligned}
$$

The End

Feedback?

Your Moment of Zen