optibus

Optimizing Vehicle and Crew Schedules in Public Transport

Daniel Rehfeldt

Public transportation optimization

- Strategic planning
  - Network design
  - Line network planning
  - ...

- Operational planning
  - Vehicle scheduling
  - Duty scheduling
  - ...

This talk considers only the most used public transport mode:

Buses

Vehicle Scheduling Optimization

Single-depot vehicle scheduling problem:

optibus

Given:

Trips $S = \{S_1, S_2 \ldots, S_n\}$.
For each trip S_i:

- t_i: departure time
- a_i: arrival time
- o_i: origin (departure terminal)
- d_i: destination (arrival terminal)

S_i	t_i	a_i	o_i	d_i
S_1	10:00	11:00	T_a	T_b
S_2	10:20	11:10	T_c	T_d
S_3	11:45	12:30	T_c	T_a
S_4	11:05	12:45	T_a	T_c

Deadhead trips (trips without passengers) of duration h_{ij} between each pair of terminals.

h_{ij}	T_a	T_b	T_c	T_d
T_a	0	30	20	25
T_b	30	0	20	10
T_c	20	20	0	15
T_d	25	10	15	0

Definition

An ordered pair of trips (S_i, S_j) is **compatible** if $a_i + h_{ij} \leq t_j$.

Single-depot vehicle scheduling problem:

Definition

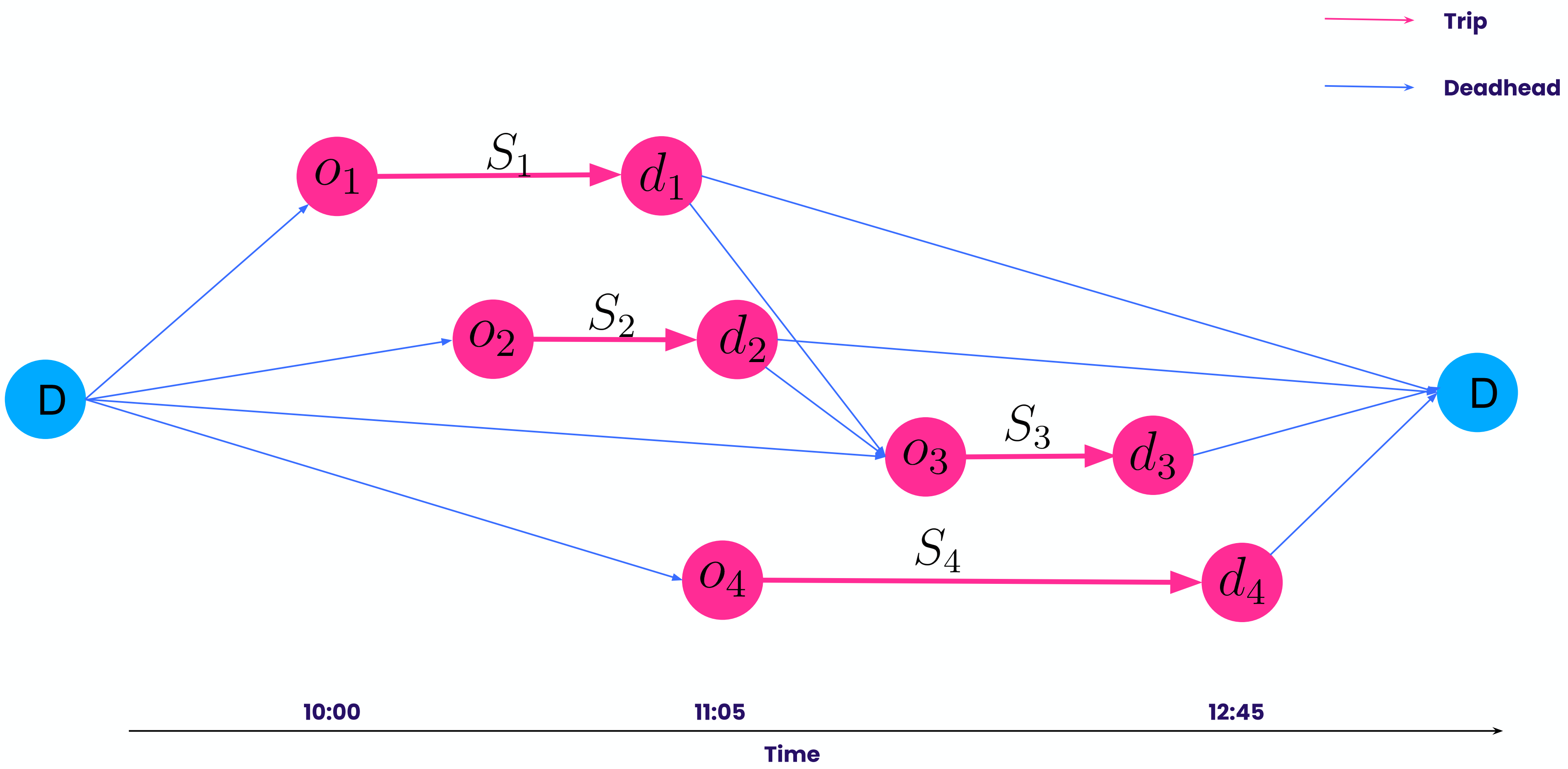
A set $B \subseteq S$ of trips is called a **vehicle block** if the elements of B can be written as a sequence $S_1, S_2, \ldots S_k$ such that each pair (S_i, S_{i+1}) is compatible.

Definition

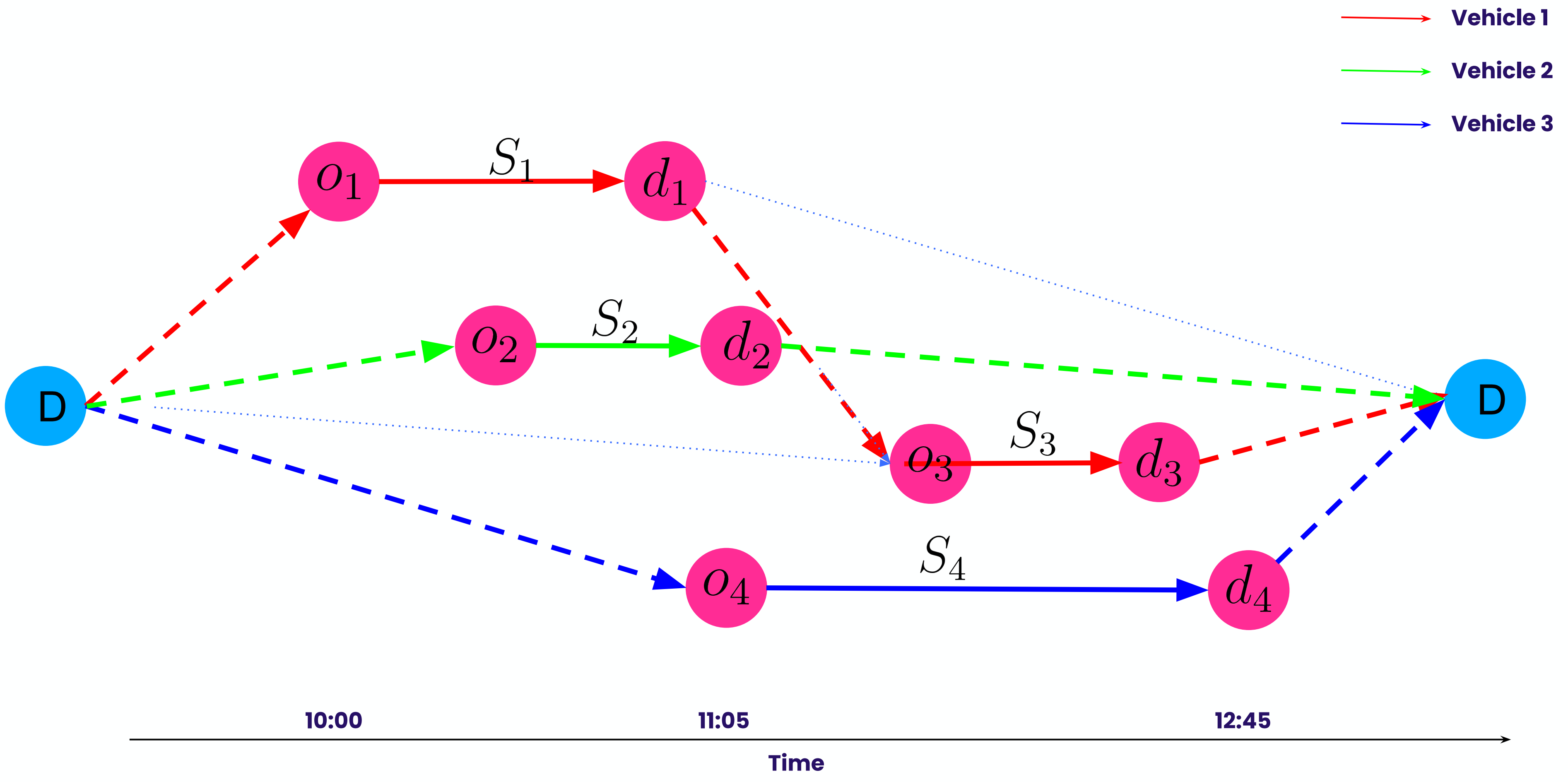
A set $\{B_1, B_2 \ldots, B_k\}$ of vehicle blocks is called a **vehicle schedule** if

$$S = \dot{\bigcup} B_i.$$

Single-depot vehicle scheduling problem (graph view):



Single-depot vehicle scheduling problem (vehicle schedule):



Single-depot vehicle scheduling problem (graph version):

Definition

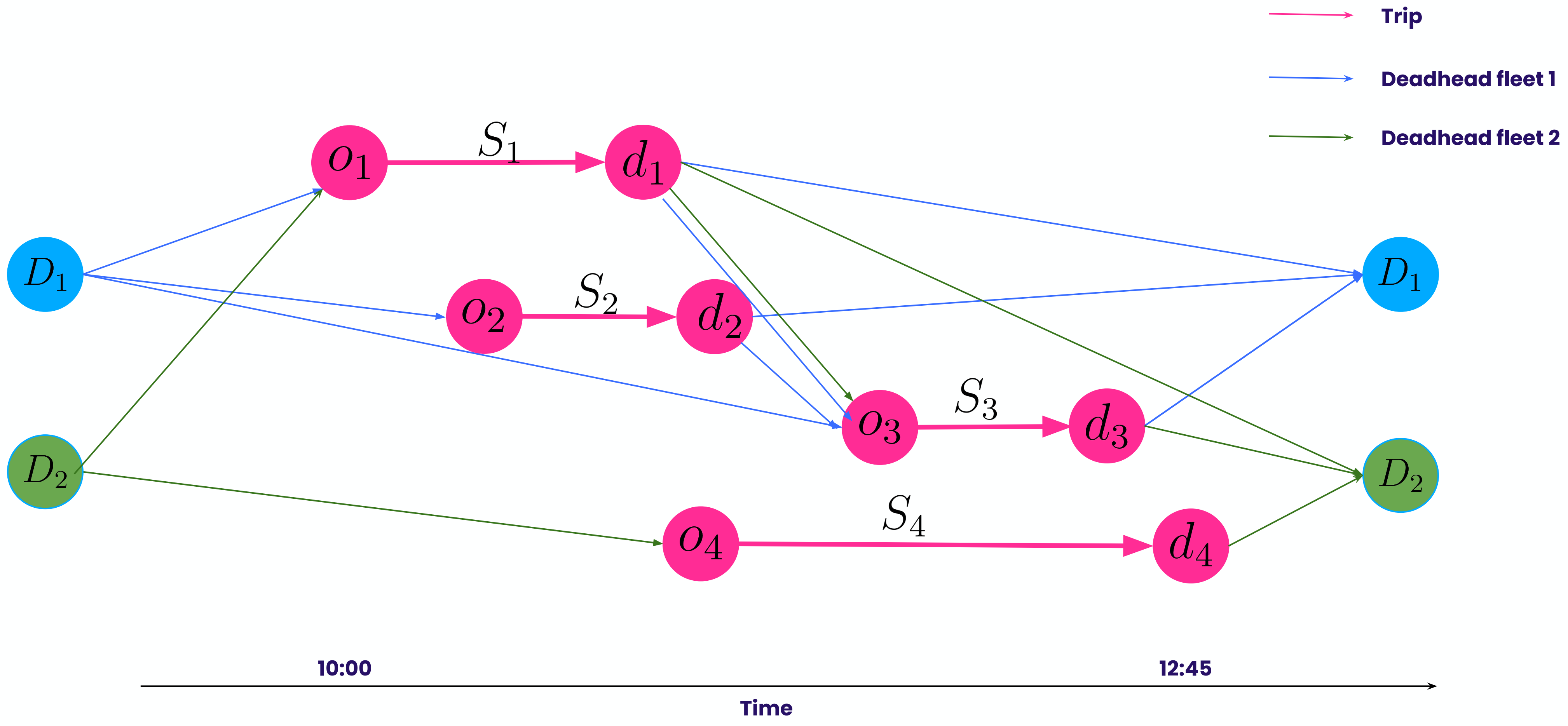
Let $D = (V, A)$ be a directed, acyclic graph with $V = T \cup \{s, t\}$ and $N^+(s) = N^-(t) = T$. Let $c \in \mathbb{R}^A_{\geq 0}$ be arc weights.

The **single-depot vehicle scheduling problem** is:

$$
\begin{array}{rlcll}
\min & c^T x & & & \\
\text{s.t.} & x(\delta^-(t)) & = & x(\delta^+(t)) & \forall t \in T \\
 & x(\delta^-(t)) & = & 1 & \forall t \in T \\
 & x(a) & \in & \{0, 1\} & \forall a \in A.
\end{array}
$$

Polynomial-time solvable (minimum cost flow problem)

Multi-depot vehicle scheduling problem (graph view):



Definition

- Let F be the non-empty set called **fleets**. Let $\kappa \in \mathbb{N}^F$, called **fleet capacities**.
- Let $D = (V, A)$ be a directed, acyclic multi-graph with $V = T \cup \{s_f, t_f : f \in F\}$ and $\delta^-(s_f) = \delta^+(t_f) = \emptyset$. Further, let $A = \dot{\bigcup}_{f \in F} A_f$, such that $\delta^+(s_f), \delta^-(t_f) \subseteq A_f$ for each $f \in F$.
- Let $c \in \mathbb{R}^A_{\geq 0}$ be arc weights.

The **multi-depot vehicle scheduling problem** is:

$$
\begin{aligned}
\min \quad & c^T x \\
\text{s.t.} \quad & x(\delta_f^-(t)) = x(\delta_f^+(t)) && \forall t \in T, \forall f \in F \\
& x(\delta^-(t)) = 1 && \forall t \in T \\
& x(\delta^+(s_f)) \leq \kappa_f && \forall f \in F \\
& x(a) \in \{0, 1\} && \forall a \in A.
\end{aligned}
$$

Multi-depot vehicle scheduling problem

optibus

The multi-depot vehicle scheduling problem is NP-hard already for two depots (Bertossi et. al., 1987).

How to solve it?

Observations:

- Usually far more variables than constraints, few variables needed -> column generation?
- Minimum cost flow subproblems (polynomially solvable)

LP relaxation can already be very hard.

Simple idea (static column generation/sifting):

Let A be the set of all columns.

1. Start with subset A’ of A, corresponding to primal feasible LP/IP solution (e.g. from heuristic)
2. Solve LP for A’
3. Compute reduced costs for remaining columns. Converged? -> stop
4. Add some columns with negative reduced cost to A’ , go to 2.

Problem: Converges very slowly due to missing path information.

Lets review an idea due to Löbel (1997):

Definition

Define Lagrangean problem $\max_\pi LR_1(\pi)$ of multi-depot vehicle scheduling problem, with

$$
\begin{aligned}
LR_1(\pi) := \min \quad & c^T x - \sum_{t \in T} \pi_t x(\delta^-(t)) + \pi^T 1 \\
\text{s.t.} \quad & x(\delta_f^-(t)) = x(\delta_f^+(t)) && \forall t \in T, \forall f \in F \\
& x(\delta^+(s_f)) \leq \kappa_f && \forall f \in F \\
& x(a) \in [0,1] && \forall a \in A.
\end{aligned}
$$

Observations:

- $LR_1(\pi)$ decomposes into $|F|$ independent single-depot problems.
- By strong duality, optimal dual solution (π, μ) to original LP provides optimal solution π to Lagrangean.

Definition

Define Lagrangean problem $\max_\mu LR_2(\mu)$ of multi-depot vehicle scheduling problem, with

$$
\begin{aligned}
LR_2(\mu) := \min \quad & c^T x - \sum_{t\in T, f\in F} \mu_t (x(\delta_f^-(t)) - x(\delta_f^+(t))) & & \\
\text{s.t.} \quad & x(\delta^-(t)) = 1 & & \forall t \in T \\
& x(\delta^+(t)) = 1 & & \forall t \in T \\
& x(\delta^+(s_f)) \leq \kappa_f & & \forall f \in F \\
& x(a) \in [0,1] & & \forall a \in A.
\end{aligned}
$$

Remarks:

- We added the (originally redundant) constraints $x(\delta^+(t)) = 1$.
- Minimum cost flow problem.

Algorithm 1:

Input: Multi-depot vehicle scheduling instance with columns A

Result: LP solution

Start with set of columns $A' \subseteq A$ that contains a primal feasible LP solution;

repeat

    Solve LP for A', LP duals: (π, μ, ρ);

    if *reduced costs indicate convergence* **then**

        return current LP solution;

    else

        Add some columns with negative reduced costs from $A \setminus A'$ to A';

        Solve $LR_1(\pi)$ and add part of primal (LP) solution to A';

        Solve $LR_2(\mu)$ and add part of primal (LP) solution to A';

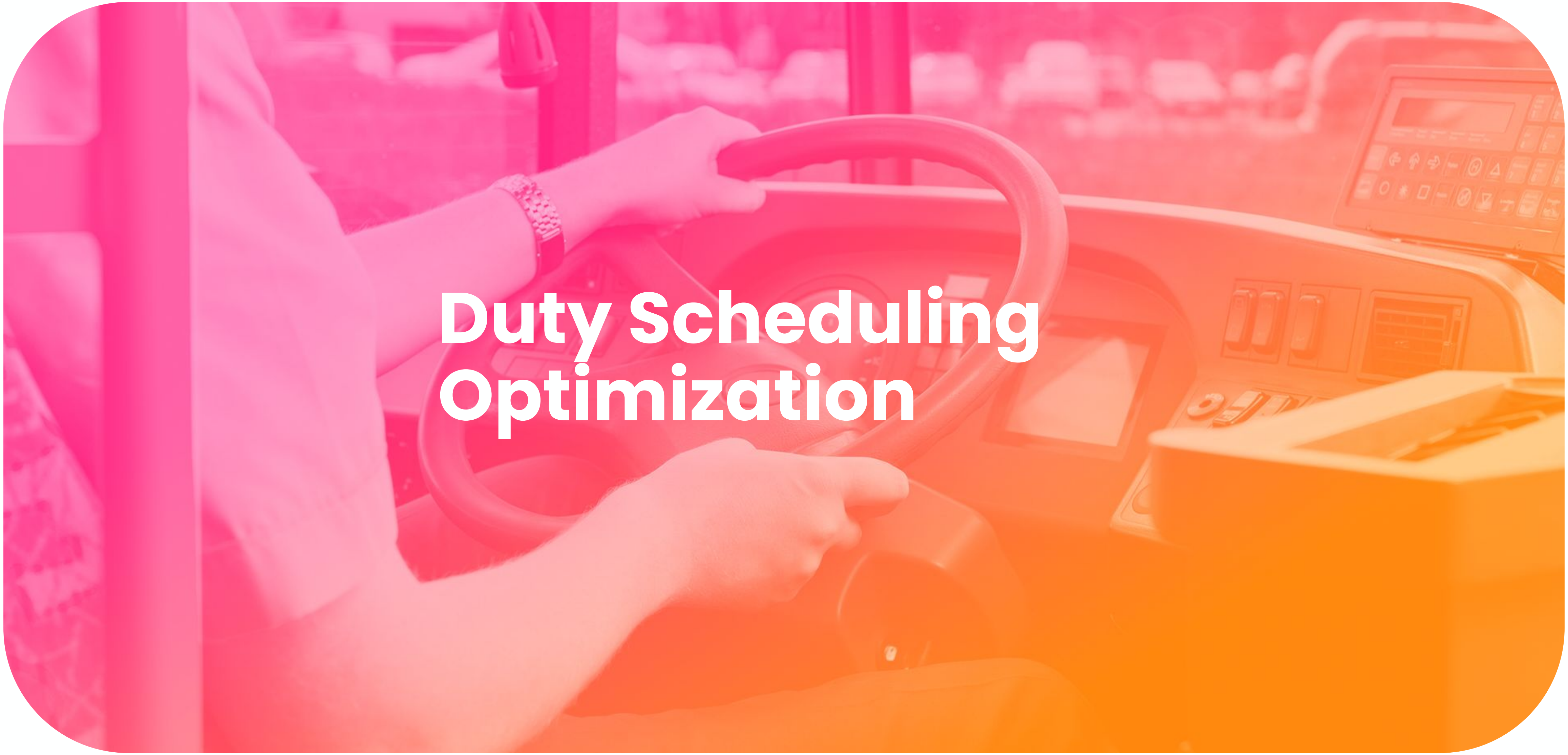
    end

until;

The algorithm doesn’t work out of the box on our vehicle scheduling problems. But with additional tricks and techniques it works well for very large cases.

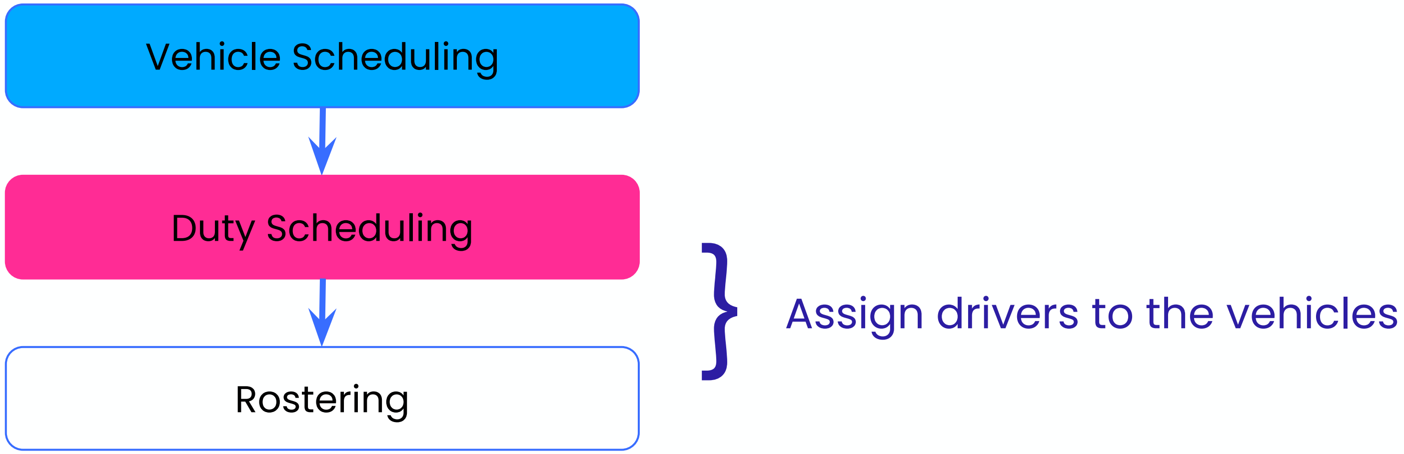
Advantages:

- Can be relatively well adapted to more complex models and model changes.
- Works even for very large problems.
- Provides LP solution with fewer fractional variables compared for example to commercial barrier solvers.

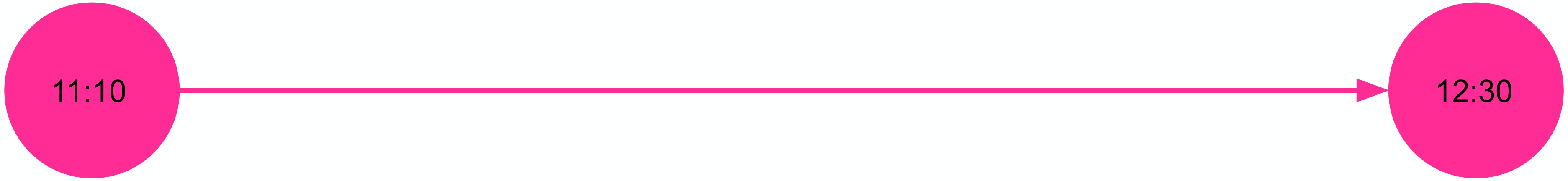


Duty Scheduling Optimization

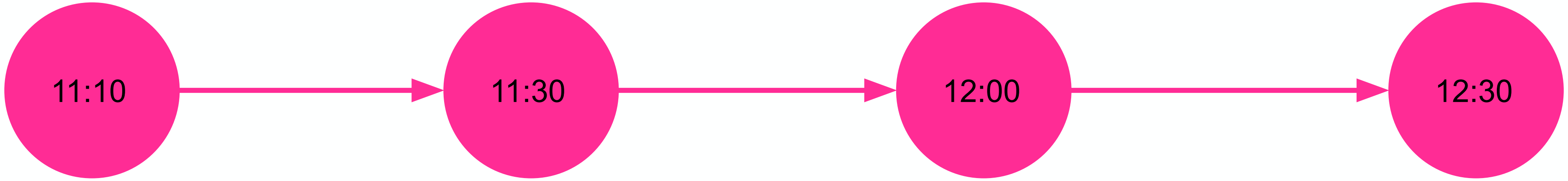
Scheduling vehicles is not enough…



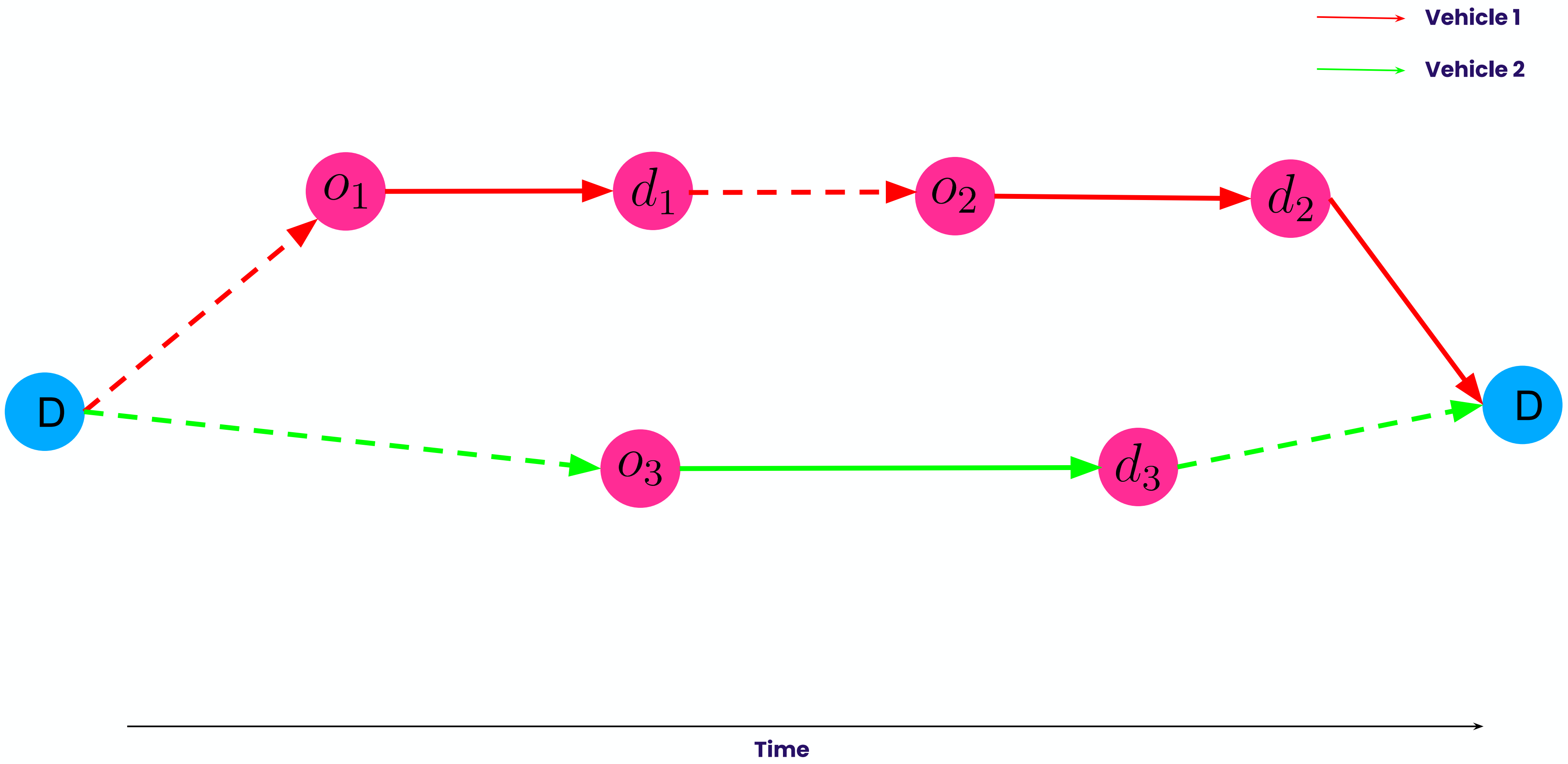
- All vehicle blocks need to be covered by duties, which follow various rules (minimum break time, maximum total duty time, etc.).
- Single trips may be covered by multiple duties.



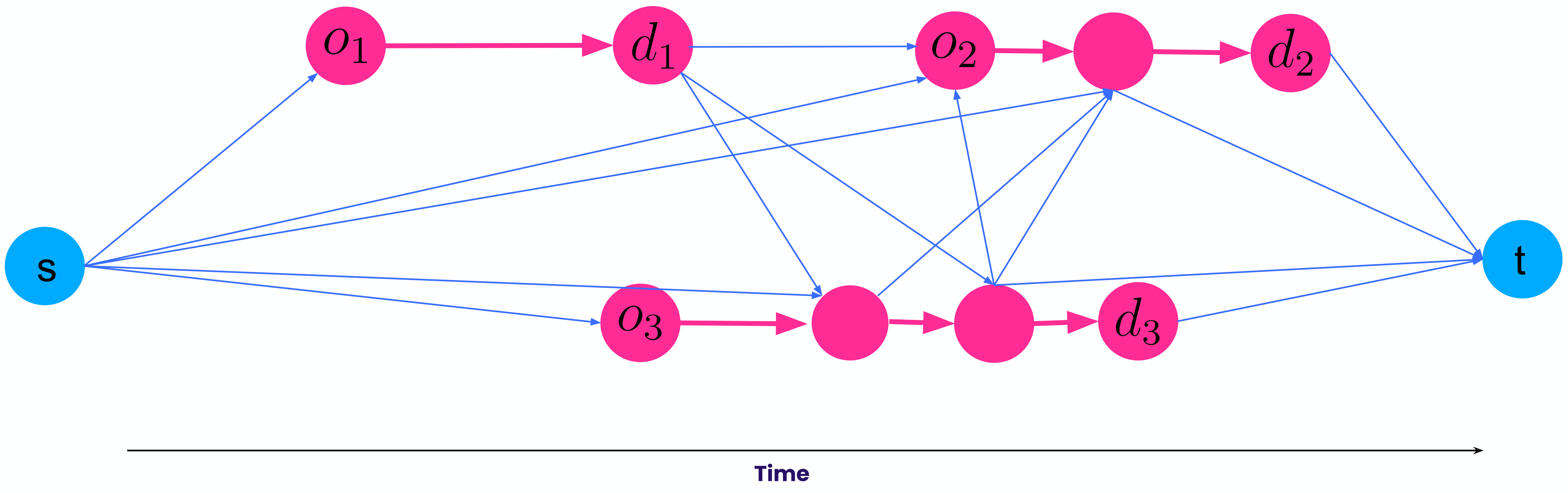
Add **relief times**, where a driver change can happen:



Duty scheduling (vehicle schedule):



Duty scheduling (duty graph with extra relief points):



Each duty is an s-t path, but not all s-t paths are duties!

Definition

- Let D be the set of duties.
- Let V_{DSP} be the set of mandatory tasks.
- Let $A \in \{0,1\}^{V_{\text{DSP}} \times D}$ be the task-duty incidence matrix.
- Let R be a matrix for additional constraints.
- Let $c \in \mathbb{R}^{D}_{\geq 0}$ be the duty costs.
- ...

The **duty scheduling problem** (Weider '07) is:

$$
\begin{aligned}
& \min c^T x + \gamma^T z, \\
\text{s.t.} \quad & Ax = \mathbf{1} \\
& Rx - z \leq r \\
& x(d) \in \{0,1\} \quad \forall d \in D \\
& z \geq 0.
\end{aligned}
$$

Duty scheduling optimization

The number of duties can be huge. Duty scheduling is typically solved by some form of column generation:

- With dynamic generation of the columns as part of the pricing
- With pre-generated columns
- ....

Combining Duty and Vehicle Scheduling

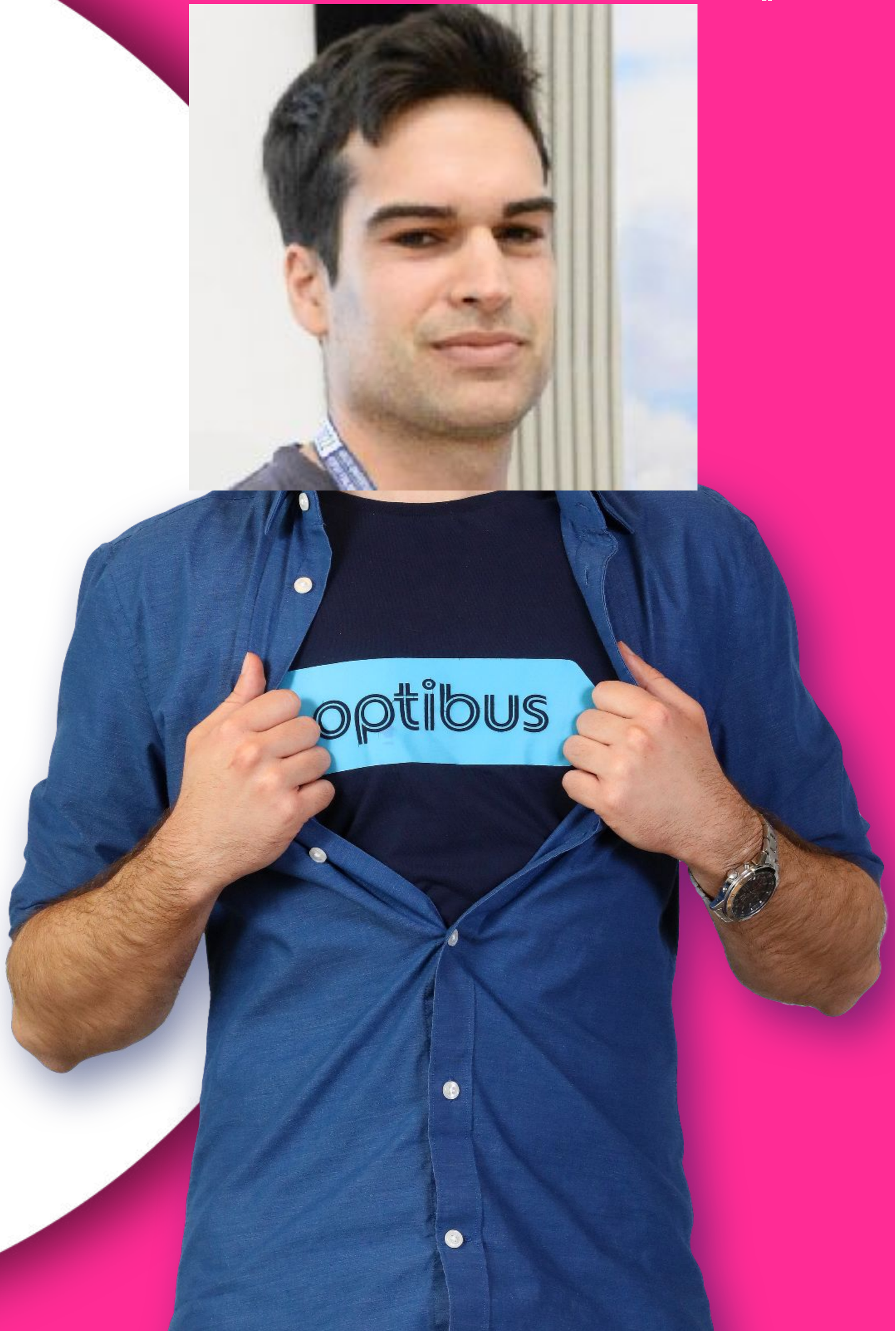
Classic approach: Compute vehicle schedule first, duty schedule second…

- …in general not globally optimal
- …might even be infeasible

Better to optimize both together.

Very hard problem and beyond the time frame of this talk.

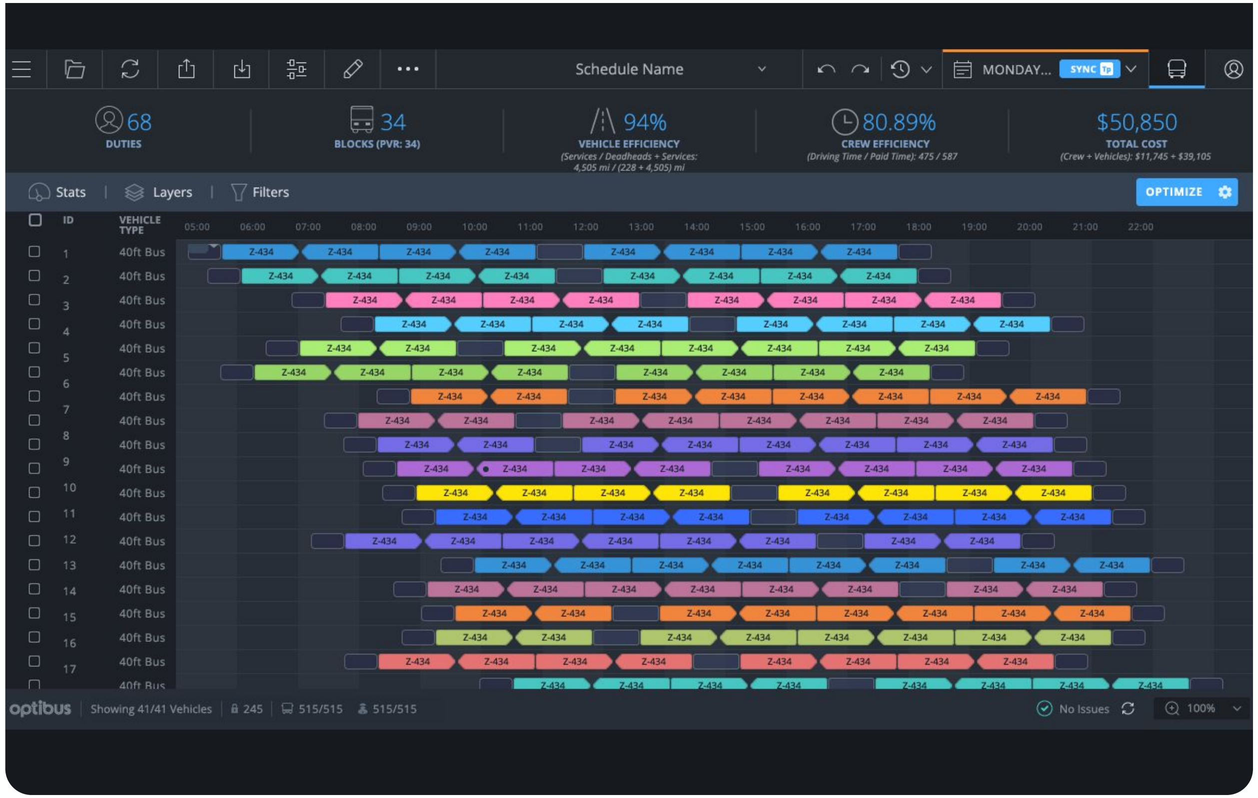
Optimization at Optibus



Optibus

- Is a SaaS company founded in 2014.
- Provides a cloud platform to solve various public transit problems.
- Optimization is a central component.
- Thousands of optimization problem instances are solved each month.

Optimization customer view



Customer can input model parameters interactively in the cloud

Algorithms run in the background

Solution (schedule) is shown, complemented by statistics and graphics

Optimization behind the scenes

From user to engine and back

Application : Vehicle and crew scheduling, time-table optimization, etc.

Schedules

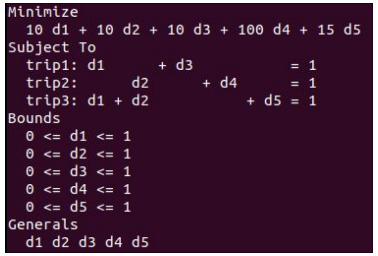
Application Data

Modelling Layer (Python) : Mathematical programming formulations.

python

Math Solution

Math Model



```
Minimize
  10 d1 + 10 d2 + 10 d3 + 100 d4 + 15 d5
Subject To
  trip1: d1      + d3           = 1
  trip2:      d2      + d4      = 1
  trip3: d1 + d2           + d5 = 1
Bounds
  0 <= d1 <= 1
  0 <= d2 <= 1
  0 <= d3 <= 1
  0 <= d4 <= 1
  0 <= d5 <= 1
Generals
  d1 d2 d3 d4 d5
```

Engine (C++) : Specialized mixed-integer programming solver.

Optimization challenges:

- Models are far more complex than shown here.
- Models can be huge, e.g. vehicle problems with tens of millions of variables.
- The integration of electric vehicles adds another layer of complexity.
- ...

Thank you

©Optibus 2024 | www.Optibus.com