**Notation**

Sets:

Parameters:

**Second Stage Formulation**

In the second stage problem, we must assign and schedule every vehicle to a charging station . To do this, the Restricted Master Problem (RMP) chooses the best routes from a set of attractive, feasible routes that the pricing subproblems will generate. A feasible route consists of every vehicle in it recharging consecutively for the exact amount of timesteps it requires to charge to its full range, with FIFO policy among the vehicles and allowing for at most one vehicle waiting in line at any given timestep. The subset of feasible routes of station we denote as and parameter states whether or not vehicle is assigned to station in route .

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | | |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

The objective function minimizes the total driving and charging cost of the system. The cost of a route of station is given by . The second stage must capture infeasibility in terms of assignments (mostly due to charging capacity constraints given by the first stage decisions), so a dummy station is included in case the RMP cannot find a feasible assignment for some vehicles. Logically, these assignments to the dummy station are penalized in the objective function. Constraints ensure that every vehicle is scheduled in one station and guarantee that every station will have as many chosen routes as chargers it has available.

The reduced cost of a route for station is:

**SPPRC Formulation for the Scheduling Problem**

We have a separate pricing subproblem for each station , and it can be modeled as a Shortest Path Problem with Resource Constraints (SPPRC) that works on a directed graph . The set of vertices is defined as , where and are dummy source and sink vertices and is the set of charging vertices for each vehicle that can be assigned to the current station, starting from the first timestep that they arrive at the station.

We define every vertex in terms of the vehicle and timestep that it represents. For the charging nodes it is straightforward to identify both attributes. For the source and sink nodes, on the other hand, we define a dummy vehicle 0 and dummy timesteps 0 and : .

The set of arcs in the graph is defined as , where:

* is the set of traveling arcs from the source node to the first charging vertex of each vehicle.
* is the set of consecutive charging arcs between the charging nodes of the same vehicle.
* is the set of arcs between charging nodes of different vehicles. This set accounts for FIFO service among the vehicles and allows connections for only the shortest waiting times between vehicles.
* is the set of travel arcs from the charging nodes to the sink node . These arcs represent the vehicles leaving the charging station.

The weights of the arcs correspond to the reduced cost along them:

**Labeling Algorithm**

Every subpath from the source node to an intermediate vertex is encoded by a label that stores the reduced cost and resource consumptions of the subpath. A label stores:

Algorithm 1 presents the label extension procedure. Lines 1 and 2 check whether the extension is feasible in terms of the recharging timesteps for the vehicle that’s recharging at , . Given an extension, if does not recharge for the exact amount of timesteps it needs to, the subpath’s corresponding label is discarded. This can happen in two different ways: undercharging (Line 1) and overcharging (Line 2). Undercharging would happen if still needs to recharge for more than one timestep and the extension leads to another vehicle starting to use the charger or to the sink node . On the other hand, overcharging would happen when has exactly one timestep left to recharge and the extension leads which would lead it to overcharge.

Additionally, Lines 3 to 5 check for feasibility in terms of the waiting line. When the given extension leads to a new vehicle starting to recharge (Line 4), we have to verify that at every timestep before there was at most one vehicle waiting in line to use the subpath’s charger. If this requirement is not complied with, the label is discarded (Line 5).

Finally, Lines 6 to 8 update the resource consumptions of the label. Line 6 updates the subpath’s reduced cost, while lines 7 and 8 update the timesteps required for the current vehicle to fully recharge, checking if the extension leads to a different vehicle or not.

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm 1:** Label extension. | | | |
|  | **Input:** | , label representing a subpath from the source node to vertex ; , arc along which the extension is performed. | |
|  | **Output:** | , label representing a subpath from the source node to vertex . | |
|  | | | |
|  | /\* check whether the extension is feasible \*/ | |  |
| **1** | **if** ∧ **then** exit | |  |
| **2** | **if** ∧ **then** exit | |  |
|  | /\* update the waiting line resource and check feasibility \*/ | |  |
| **3** | **for** **do** | |  |
| **4** | **if** **then** | |  |
| **5** | **if** **then** exit | |  |
|  | /\* update the resource consumptions \*/ | |  |
| **6** |  | |  |
| **7** | **if** **then** | |  |
| **8** | **else** | |  |
| **9** | **return** | |  |