


Hydrogen bus route planning in regional Victoria

R. Esmailbeigi ^a  V. Mak-Hau ^a  G. Pineda-Villavicencio ^a  and J. Ugon ^a 

^a*School of Information Technology, Deakin University, Geelong, VIC 3218, Australia*

Email: r.esmailbeigi@deakin.edu.au

Abstract: In this paper, we present a case study to optimise refuelling logistics for a hydrogen bus network in regional Victoria, Australia. The bus network under study is operated by a company that aims at introducing a new fleet of hydrogen buses. We use integer linear programming to design a decision support system that could guide the company to determine the optimal configuration of a hydrogen refuelling network. Hydrogen bus refuel speed is governed by the pressure of the refuelling pumps. A high-pressure refuel pump is more costly but has a faster refuel time, whereas a low-pressure pump is relatively more economical, but it has a longer refuel time. Higher pressure storage also allows larger volumes of hydrogen to be stored at refuel points, potentially lowering the bulk refilling frequency. As with the refuelling speed, higher pressure storage vessels cost significantly more than standard pressure storage vessels. So there is a trade-off between investing in expensive high-pressure refuel equipment so buses can be refuelled in the shortest practicable time, and in lower-cost and low-pressure refuel equipment resulting in longer refuel times. Furthermore, four replenishment activities (i.e., refuelling, exterior cleaning, internal cleaning, and potentially COVID-19 cleaning) need to be scheduled for each bus. Results of our case study show that it is possible to satisfy the total demand using one low-pressure refuelling pump. In addition, we showed that without our proposed decision support tool, the company would use 9.1% more vehicles compared to the optimal number of buses.

Keywords: *Bus scheduling, clean transport, decision support system, logic-based Benders decomposition*

1 INTRODUCTION

Public transit around the world is undergoing a shift towards cleaner and more efficient vehicles. This shift to new types of fuel requires planners to rethink how to operate their network, introducing a need for new software solutions to accommodate the new technology. A company operating in a rural town in Australia is introducing hydrogen buses to entirely replace their fleet. The purpose of this paper is to present the mathematical model and software solution that we developed to assist the company towards this transition. This model is translatable to other contexts in public transportation networks where new fleets of buses are introduced.

Hydrogen fueled vehicles differ from conventional vehicles in the way the fuel is produced, transported, stored, converted and applied (Singla *et al.* 2021), with storage and pumping of particular importance to public transport companies. A model for operating a hydrogen-fueled fleet must take into account such storage and refueling requirements. Optimising aspects of the running of bus fleets has been considered in the literature before. A frequent request has been the design of scheduling of bus routes. Wagale *et al.* (2013) presented a model that optimises operating costs of a bus service network while incorporating several real time data such as bus stop departures and arrivals. The model is an adaptation of an earlier model of Kim *et al.* (2009), and it has been applied to the Jaipur City Transport Service Limited. Kang *et al.* (2019) instead proposed three integer linear programming (ILP) models to formulate the bus driver scheduling, bus and driver scheduling, and bus and driver scheduling with mealtime windows, respectively. It is worth noting that there exist models that combine the driver rostering and vehicle scheduling problems, however, in general the computation time is long. Haghani & Banihashemi (2002) presented new models for the multiple depot vehicle scheduling problem and the multiple depot vehicle scheduling problem with route time constraints. The resulting model is compared to the 1998 schedules of the mass transit administration (MTA) in Baltimore (Banihashemi 1998), and the comparison indicates that the proposed model improves upon the MTA schedules in all respects.

More recently, there has been much emphasis on optimising the operation of electric buses (EBs). The principal methodology is mixed-integer optimisation, and the general aim is to minimise the total cost of operating fleets of electric buses, which may include regular operation, maintenance, and refueling. We mention four papers in this area. Alwesabi *et al.* (2020) developed a mixed-integer model that simultaneously selects the optimal location of dynamic wireless charging facilities and finds the optimal battery sizes of electric buses for a transportation network. In the model, the EBs are required to travel their common routes despite their different optimal battery sizes. In addition, the authors developed a scheduling model to find the optimal number of EBs; here an EB can serve different routes, but the battery size should be enough to serve any of the routes in the transportation network.

Another model for electric buses was proposed by Rogge *et al.* (2018), which aims to minimise the total cost of ownership consisting of vehicle investment, charger investment, operational costs, and energy expenses. The framework is based on a grouping genetic algorithm that incorporates a mixed-integer charger optimisation. Wang *et al.* (2017) designed another model that aims to minimise the total cost of ownership. It is a mixed-integer optimisation model for optimising electric bus recharging schedules. The model determines both the planning and operational decisions while minimising total annual costs. These decisions include the number of chargers to be installed and recharging scheduling. This model was applied to a real-world transit network based in Davis, California. Finally, Dirks *et al.* (2021) also proposed a mixed-integer model that minimises the total cost of ownership of electric buses. In addition, the paper assesses emission reductions of nitrogen oxide that are achieved by an integration of EBs. In the recent paper by Golla *et al.* (2021), the authors present a model for scheduling and operating hydrogen buses with on-site electrolysis, and test it on a fleet of 24 buses. Their focus is on meeting the maximum peak demand.

We employ integer linear programming for scheduling four replenishment activities (i.e., refuelling, exterior cleaning, internal cleaning, and potentially COVID-19 cleaning) for a set of hydrogen buses that need to serve a given pool of bus trips. We apply the model to a case of hydrogen bus route planning in regional Victoria, Australia. Our proposed solution guides our industry partner in determining an efficient configuration of their hydrogen refuelling network. A relevant consideration is the pressure of the refuelling pumps, the higher the pressure of the pump, the higher the cost of it, and the lower the refuelling time. Using high pressure pumps also increases the risk of a disaster Sakamoto *et al.* (2018).

We solve the problem through a logic-based Benders decomposition approach (Hooker & Ottosson 2003). In our implementation, Benders cuts are generated within a single branch and bound search tree. This strategy

is often referred to as Branch and Check (B&CH) in the literature (Thorsteinsson 2001, Beck 2010). The potential competitiveness of B&CH for solving a variety of combinatorial optimisation problems has been discussed in Esmailbeigi *et al.* (2021). Results of our case study show that it is possible to satisfy the total demand using one low-pressure refuelling pump. In addition, we showed that without our proposed decision support tool, the industry partner would use 9.1% more vehicles compared to the optimal number of buses. Our proposed methodology prescribes a solution within a few seconds.

In the next subsection we introduce the problem in details. The rest of the paper is organised as follows. In Section 2, we describe our solution methodology. In Section 3, we provide implementation details and results of the case study. Finally we conclude the paper by discussing future research directions.

1.1 Problem description

The purpose of this paper is to schedule a fleet of buses to cover a number of trips, in the context of upgrading the network from an existing fleet of fossil-fuelled buses to a fleet of hydrogen-fueled vehicles. The trips follow pre-defined routes according to a pre-defined timetable. The network of routes is already established, as are the trip timetables (start and end time of each trip), and the problem is to assign buses to each of the routes, allowing for refueling and cleaning times.

We present a number of definitions at the outset. A **bus trip** is defined by a start location, a departure time, a destination, and an arrival time, all given as parameters to the problem. If the start location and the destination are the same, the trip is a loop. A **terminal** is a bus stop that is either a start or an arrival location for a bus trip. A **bus route** is a collection of bus trips between two specific locations A and B (either with A as start location and B the destination, or vice versa) or a collection of loops that begin and end at the same location. A **bus schedule** is defined by a sequence of bus trips to be run by a bus. These are the business solutions to be determined by the proposed methodology.

An **ordinary connection** of two bus trips (Trip 1 and Trip 2) is one in which the difference between the end time of Trip 1 and the start time of Trip 2 is enough for a trip from the destination terminal of Trip 1 to the departure terminal of Trip 2. There are currently four **replenishment activities** (or, tasks) that must be performed exactly once every day. They are: refuelling (typically takes 10 minutes if high-pressure pumps are used or 20 minutes if low-pressure pumps are used), exterior washing (30 minutes), interior mopping and cleaning (20 minutes), and COVID-19-cleaning (25 minutes). A **replenishment connection** of two bus trips (Trip 1 and Trip 2) is one where there is sufficient time for one or more of the four replenishment activities to be carried out, together with the time required to travel from the destination of Trip 1 and the replenishment location and the time required to travel between the replenishment location to the start location of Trip 2. The **replenishment capacity** is the maximum number K of buses that can be refilled simultaneously at the replenishment location. Furthermore, at most M buses can be undergo any cleaning activity at the same time.

Each bus is to be refuelled and cleaned (including exterior, interior, and COVID-19) exactly once a day. Let $D = (V^*, A^*)$ be a digraph that represents the bus scheduling problem with $V^* = \{source, V, sink\}$, $V = \{1, \dots, n\}$ a set of intermediates nodes each representing a bus trip. Note that *source* and *sink* are dummy/auxiliary nodes in a network where all buses respectively start and end their trips. Arcs A^* of the network represent possible connections between two bus trips. Specifically, each arc in A^* represents a feasible connection between two bus trips, for example, between Bus Trips 1 and 2, such that the difference between the end time of Bus Trip 1 and the start time of Bus Trip 2 is enough for a trip from the destination terminal of Trip 1 to the departure terminal of Trip 2. Furthermore we also define the set of **replenishment arcs** ($\mathcal{R} \subset A^*$) when the difference between the end time of Bus Trip 1 and the start time of Bus Trip 2 is enough for a trip from the destination terminal of Trip 1 to the refuelling/cleaning hub, get refuelled/cleaned, and then travel to the departure terminal of Trip 2.

Figure 1 shows some ordinary (black) and replenishment (green) arcs and two candidate bus routes (bold arcs). In this figure, each route/path corresponds to one bus. The maximum number of replenishment activities that can be performed on Arc a , denoted c_{ij} , determines whether it is a replenishment arc ($c_{ij} > 0$) or an ordinary arc ($c_{ij} = 0$). We also use \mathcal{B} to denote the set of buses and \mathcal{T} to denote the set of time slots per day. The duration of each time slot is considered as 1 minute, resulting in a total of $|\mathcal{T}| = 1440$ time slots.

Our industry partner is currently interested in a feasible set of bus schedules in which: 1) all trips are covered; 2) all replenishment activities are performed for each bus once a day; 3) all buses are refueled once a

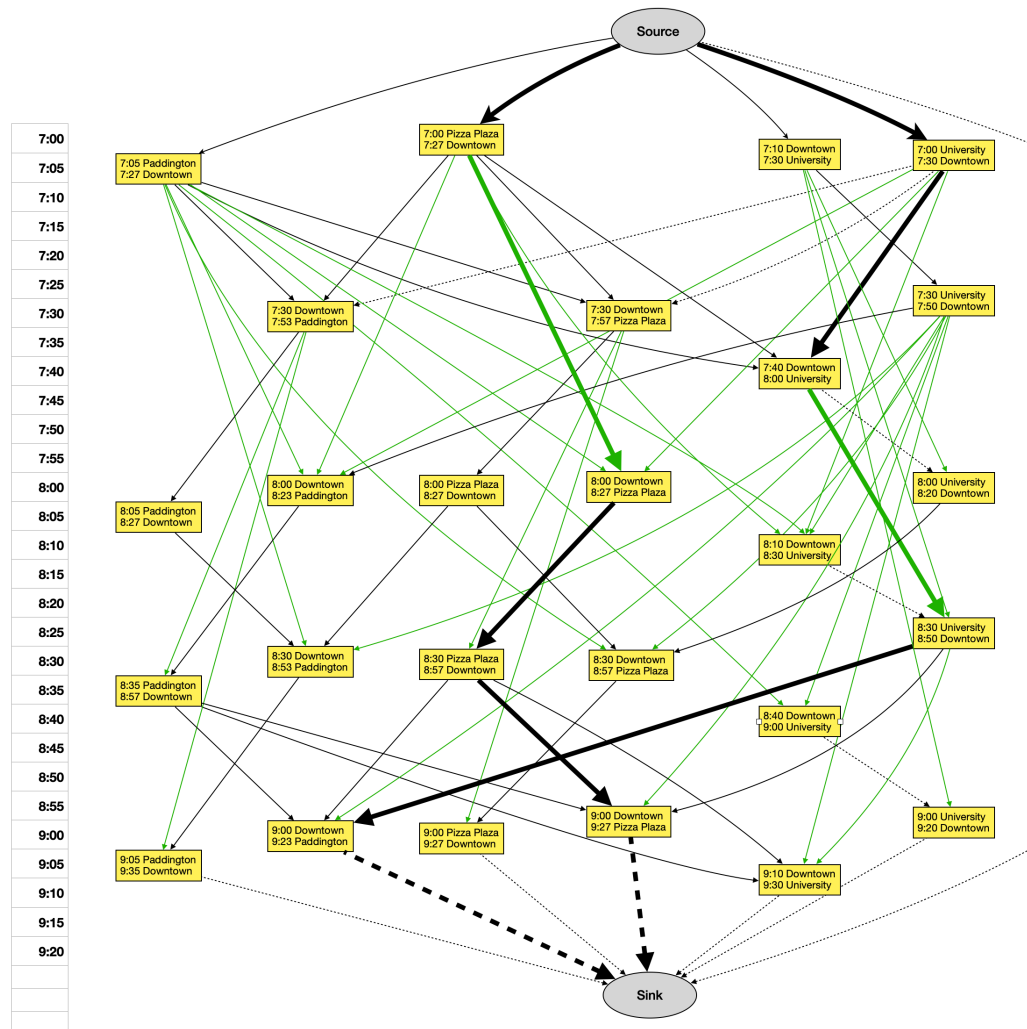


Figure 1. An example of two bus routes on the network (each route corresponds to one bus). Each node on the graph (yellow boxes) corresponds to a bus route. The grey ellipses are the source and sink nodes representing the start and end of the day respectively. Two nodes are connected by a directed arc when it is possible for a bus to perform both routes. When there is time for replenishment between the routes the arc is green. Sequences of bold arcs represent possible daily bus schedules (dashed arcs show when the bus has ended its daily schedule and is returned to the depot.)

day; 4) the replenishment and refuelling capacities are not violated at any time; and 5) minimum number of buses are used, in which case, the remaining bus(es) can be used as backups.

2 THE PROPOSED METHODOLOGY

In order to solve the problem, we need to perform two tasks: assign trips to buses, so that no simultaneous trips are assigned to any bus, allowing for replenishment time in the schedule, and then scheduling replenishment and cleaning times in agreement with the bus schedules. We employ B&CH to solve the problem. We begin by introducing the master problem and subproblem which constitute the proposed B&CH. The master problem finds candidate bus routes. The subproblem receives these routes as an input, and schedules replenishment activities for them, if possible. If it is found that the master solution induces an infeasible subproblem, a Benders infeasibility cut is added to the master problem to remove the infeasible solution, and the search continues until a feasible solution is obtained.

2.1 The Master Problem

Let x_{ij} be a binary decision variable with $x_{ij} = 1$ if the arc (i, j) is used in the solution (i.e., the bus will take Bus Trip j immediately after Bus Trip i), and $x_{ij} = 0$ otherwise. The objective of the master problem is to maximise the total time when buses are available for replenishment.

$$\max \sum_{(i,j) \in \mathcal{R}} c_{ij} x_{ij} \quad (1)$$

$$\text{s.t.} \sum_{j \in V} x_{\{source\},j} = |\mathcal{B}| \quad (2)$$

$$\sum_{i \in V} x_{i,\{sink\}} = |\mathcal{B}| \quad (3)$$

$$\sum_{(i,j) \in A^*} x_{ij} = 1, \quad \forall j \in V \quad (4)$$

$$\sum_{(j,i) \in A^*} x_{ji} = 1, \quad \forall j \in V \quad (5)$$

The objective function (1) maximises the total amount of time buses are available for replenishment during their schedules. Constraints (2) and (3) require that $|\mathcal{B}|$ paths leave the source node and enter the sink node (note that there is an arc from the source to every vertex in V , and an arc from every vertex in V to the sink), ensuring that a schedule is constructed for each bus. Constraints (4) and (5) guarantee that every bus trip will be covered by one bus.

2.2 The Subproblem

Each path obtained from the master solution represents a schedule for a bus. This solution provides us with a set of time windows for each bus, where the replenishment activities can take place. The subproblem utilises this data and seeks to schedule the replenishment activities for all buses at the same time.

Let $\mathcal{T}_b \subset \mathcal{T}$ be the set of time slots available for Bus $b \in \mathcal{B}$ for performing the replenishment activities. Although the subproblem is a feasibility problem, we define an objective function to maximise the likelihood of scheduling replenishment activities together (in one replenishment arc) so the buses do not have to travel to the replenishment hub too often (hence saving fuel). To this end, we define π_{bt} as the potential benefit of scheduling a replenishment activity for Bus $b \in \mathcal{B}$ at time $t \in \mathcal{T}_b$. We set this parameter to the capacity of the arc to which t belongs. We also use $\Omega = \{R, E, I, C\}$ to denote the set of replenishment activities: refuelling (R), exterior cleaning (E), internal cleaning (I), and COVID-19 cleaning (C). The duration of the replenishment activity $r \in \Omega$ is denoted by d_r .

The subproblem employs the following decision variables: $z_{bt}^r \in \{0, 1\}$, for $b \in \mathcal{B}$, $r \in \Omega$, and $t \in \mathcal{T}_b$ is a binary decision variable with $z_{bt}^r = 1$ indicating Bus b commences replenishment activity r in time-slot t ; and $y_{bt}^r \in \{0, 1\}$, for $b \in \mathcal{B}$, $r \in \Omega$, and $t \in \mathcal{T}_b$ is a binary decision variable with $y_{bt}^r = 1$ indicating Bus b is undergoing replenishment activity r in time-slot t . Next we introduce the subproblem. Recall that the index b represents a known bus route, and therefore there will be no issue of symmetry in the formulation.

$$\max \sum_{b \in \mathcal{B}} \sum_{t \in \mathcal{T}_b} \pi_{bt} \sum_{r \in \Omega} z_{bt}^r \quad (6)$$

$$\text{s.t.} \sum_{\delta=0}^{d_r-1} z_{b,t-\delta}^r = y_{bt}^r \quad \forall r \in \Omega, b \in \mathcal{B}, t \in \mathcal{T}_b \quad (7)$$

$$\sum_{t \in \mathcal{T}_b} z_{bt}^r = 1 \quad \forall b \in \mathcal{B}, r \in \Omega \quad (8)$$

$$\sum_{t \in \mathcal{T}_b} y_{bt}^r = d_r \quad \forall b \in \mathcal{B}, r \in \Omega \quad (9)$$

$$\sum_{r \in \Omega} y_{bt}^r \leq 1 \quad \forall b \in \mathcal{B}, t \in \mathcal{T}_b \quad (10)$$

$$\sum_{b \in \mathcal{B}: t \in \mathcal{T}_b} \sum_{r \in \Omega \setminus \{R\}} y_{bt}^r \leq M \quad \forall t \in \mathcal{T} \quad (11)$$

$$\sum_{b \in \mathcal{B}: t \in \mathcal{T}_b} y_{bt}^R \leq K \quad \forall t \in \mathcal{T} \quad (12)$$

The objective function (6) rewards scheduling of the replenishment activities in the described arcs. Constraints (7)–(9) ensure that a replenishment activity for each bus is performed once and in consecutive time slots. Constraint (10) requires that no more than one replenishment activity is carried out for a bus at the same time. Constraints (11) and (12) respectively enforce the cleaning and refuelling capacities.

If the subproblem is infeasible, then it means that with the current set of replenishment arcs used in the master solution, not all buses can fulfil all replenishment activities. In this case, a Benders feasibility cut is added to the master problem to eliminate the current solution. Let X^* denote the index set of x variables in the current master solution that have a value of 1. We add the following nogood cut to the master problem:

$$\sum_{(i,j) \in A^* \setminus X^*} x_{ij} + \sum_{(i,j) \in X^*} (1 - x_{ij}) \geq 1 \quad (13)$$

2.3 Performance Boost

In addition to the violation of the replenishment capacities, the master solution will be infeasible if at least one of the resulting $|\mathcal{B}|$ routes is infeasible with respect to the total time windows available for performing the four replenishment activities. In order not to solve the infeasible subproblems frequently, we employed a procedure for checking feasibility of each route before solving the subproblem. The algorithm proceeds to solving the subproblem only if this preliminary check is passed.

Consider a replenishment arc between Trip 1 and Trip 2. We define the free time of this replenishment arc as the difference between the end time of Trip 1 and the start time of Trip 2 minus the time it takes to travel from the destination of Trip 1 to the facility and from the facility to the departure of Trip 2, assuming that the replenishment activity with the minimum duration is performed in the facility. We also define the total free time of a route as the summation of the free times of its replenishment arcs. If the total time needed for completing the four replenishment activities exceeds the total free time of a route, then the route (and so the master solution) is infeasible. We add a feasibility cut as soon as one infeasible route is found.

3 IMPLEMENTATION DETAILS AND RESULTS

The computer used to test the optimiser is a Dell Optiplex core i5-4670 with 3.40GHz processors and 24GB RAM running Ubuntu 18.04. We used Gurobi 9.1 as the optimisation solver and employed a lazy callback to implement the proposed B&CH approach. All solver parameters were set to their default values except for the `SolutionLimit` parameter which was set to 1 (the master problem stops as soon as a feasible solution is found). As a result, B&CH terminates as soon as the subproblem returns a feasible solution. The resulting network consists of 207 nodes and 18,718 arcs (17,548 replenishment arcs and 1,170 ordinary arcs).

We originally used the total free time of arcs in the objective function of the master problem instead of using their capacities. In that case, we observed that it would take hundreds of thousands of feasibility cuts before the algorithm terminates due to an out-of-memory issue after nearly 4 hours. In fact, all feasibility cuts were added based on the preliminary feasibility check introduced in Section 2.3 (not even a single subproblem is solved). Using the present objective function, a feasible solution is obtained after a few iterations (within a few seconds).

We experimented scenarios with $M = 2$, $K \in \{1, 2\}$ and both low-pressure and high-pressure pumps as per the advice of our industry partner. We found that it is possible to satisfy the total demand using one low-pressure refuelling pump. Furthermore, without our proposed decision support tool, the company would use 9.1% more vehicles compared to the optimal number of buses.

4 CONCLUSIONS

In this study, we employed integer linear programming with branch and check for route planning of a set of hydrogen buses. Results of our case study demonstrated the significance of utilising smart decision support systems for making managerial decisions. Our analysis informs our industry partner about a cost-saving feasible solution even in the worst-case scenario when only one low-pressure pump is available. The company

would not be able to solve such a combinatorial optimization problem manually. Incorporating more realistic assumptions into the problem constitutes an important part of the future research in this area. A fully-fledged decision support system could provide solutions that help determine the number, type, capacity and location of multiple refuelling or storage facilities. Of interest is to evaluate the implications of considering mobile refuelling vehicles in the efficacy and resilience of such operational plans. These aspects can also be integrated with driver rostering decisions.

Acknowledgements: This project is funded by Deakin Hycel Technology Hub. We are grateful to our industry partner for providing us with numerical data and the information needed to formulate the problem.

REFERENCES

- Alwesabi, Y., Wang, Y., Avalos, R. & Liu, Z. (2020), 'Electric bus scheduling under single depot dynamic wireless charging infrastructure planning', *Energy* **213**, 118855.
- Banihashemi, M. (1998), Multiple depot transit scheduling problem considering time restriction constraints, PhD thesis, Ph. D. dissertation, Department of Civil Engineering, University of Maryland
- Beck, J. C. (2010), Checking-up on branch-and-check, in 'International Conference on Principles and Practice of Constraint Programming', Springer, pp. 84–98.
- Dirks, N., Schiffer, M. & Walther, G. (2021), 'On the integration of battery electric buses into urban bus networks', *arXiv preprint arXiv:2103.12189* .
- Esmailbeigi, R., Mak-Hau, V., Yearwood, J. & Vivian, N. (2021), 'The multiphase course timetabling problem', *European Journal of Operational Research* pp. 1–35. <https://doi.org/10.1016/j.ejor.2021.10.014>.
- Golla, A., vom Scheidt, F., Röhrig, N., Staudt, P. & Weinhardt, C. (2021), *Vehicle Scheduling and refueling of Hydrogen Buses with On-site Electrolysis*, Gesellschaft für Informatik, Bonn. Accepted: 2021-01-27T13:34:15Z.
- Haghani, A. & Banihashemi, M. (2002), 'Heuristic approaches for solving large-scale bus transit vehicle scheduling problem with route time constraints', *Transportation Research Part A: Policy and Practice* **36**(4), 309–333.
- Hooker, J. N. & Ottosson, G. (2003), 'Logic-based Benders decomposition', *Mathematical Programming* **96**(1), 33–60.
- Kang, L., Chen, S. & Meng, Q. (2019), 'Bus and driver scheduling with mealtime windows for a single public bus route', *Transportation Research Part C: Emerging Technologies* **101**, 145–160.
- Kim, W., Son, B., Chung, J.-H. & Kim, E. (2009), 'Development of real-time optimal bus scheduling and headway control models', *Transportation Research Record* **2111**(1), 33–41.
- Rogge, M., van der Hurk, E., Larsen, A. & Sauer, D. U. (2018), 'Electric bus fleet size and mix problem with optimization of charging infrastructure', *Applied Energy* **211**, 282–295.
- Sakamoto, J., Misono, H., Nakayama, J., Kasai, N., Shibutani, T. & Miyake, A. (2018), 'Evaluation of safety measures of a hydrogen fueling station using physical modeling', *Sustainability* **10**(11), 3846.
- Singla, M. K., Nijhawan, P. & Oberoi, A. S. (2021), 'Hydrogen fuel and fuel cell technology for cleaner future: a review', *Environmental Science and Pollution Research* **28**(13), 15607–15626.
- Thorsteinsson, E. S. (2001), Branch-and-Check: A hybrid framework integrating mixed integer programming and constraint logic programming, in 'International Conference on Principles and Practice of Constraint Programming', Springer, pp. 16–30.
- Wagale, M., Singh, A. P., Sarkar, A. K. & Arkatkar, S. (2013), 'Real-time optimal bus scheduling for a city using a dtr model', *Procedia-Social and Behavioral Sciences* **104**, 845–854.
- Wang, Y., Huang, Y., Xu, J. & Barclay, N. (2017), 'Optimal recharging scheduling for urban electric buses: A case study in davis', *Transportation Research Part E: Logistics and Transportation Review* **100**, 115–132.