Public Transport Network Design for Equality of Accessibility via Message Passing Neural Networks and Reinforcement Learning

Duo Wang[a], Andrea Araldo[b] and Maximilien Chau
SAMOVAR, Institut Polytechnique de Paris, Palaiseau, France
{duo-wang, andrea.araldo}@telecom-sudparis.eu, maximilien.chau@ip-paris.fr

Keywords: Transport, Network, Design, Accessibility, Reinforcement Learning, Message Passing Neural Networks.

Abstract: Graph learning involves embedding relevant information about a graph's structure into a vector space. However, graphs often represent objects within a physical or social context, such as a Public Transport (PT) graph, where nodes represent locations surrounded by opportunities. In these cases, the performance of the graph depends not only on its structure but also on the physical and social characteristics of the environment. Optimizing a graph may require adapting its structure to these contexts. This paper demonstrates that Message Passing Neural Networks (MPNNs) can effectively embed both graph structure and environmental information, enabling the design of PT graphs that meet complex objectives. Specifically, we focus on accessibility, an indicator of how many opportunities can be reached in a unit of time. We set the objective to design a "equitable" PT graph with a lower accessibility inequality. We combine MPNN with Reinforcement Learning (RL) and show the efficacy of our method against metaheuristics in a use case representing in simplified terms the city of Montreal. Our superior results show the capacity of MPNN and RL to capture the intricate relations between the PT graph and the environment, which metaheuristics do not achieve.

1 INTRODUCTION

Existing Public Transport (PT) is less and less adequate to satisfy mobility needs of the people, in a context of urban sprawl (Sun et al., 2018). The United Nations estimate that only "1/2 of the urban population has convenient access to PT" (UN, 2020). Building more and more PT lines to keep pace with urban sprawl, using traditional planning objectives, has proved to be ineffective.

PT operators generally design PT lines with the purpose of maximizing overall efficiency, measured in terms of generalized cost (which takes into account travel times and cost for the operators), or number of kilometers traveled or number of passengers transported. This has resulted in unequal development of PT within urban areas. The level of service offered by PT is often satisfactory in city centers and poor in the suburbs. In transportation, there is a consensus that it is not sustainable that people in the suburbs travel by car, as private cars are the largest polluters, 60.6% of all transport (EU, 2019). Therefore, suburban population depend on their private cars to perform their daily activities (Anable, 2005; Welch et al., 2013). As an example, the modal share of the car in the city center of Prague is double that in the city center. The dependence on private cars has negative economic, social and environmental impacts (Saeidizand et al., 2022, Section 2.2), which are common to different cities of the world. For example, 61% of EU road transport CO_2 comes from cars, jobseekers with no car have 72% less chances of finding a job in Flanders, etc. Therefore, a sufficient condition to achieve sustainability is to improve PT level of service where it is currently poor. We propose in this paper to set geographical equality of PT level of service as the main design objective. We focus in this paper on PT *accessibility* metric, which measures the ease (in terms of time and/or monetary cost) of reaching Points of Interest (PoIs) via PT. To improve geographical equality, we prioritize increasing PT accessibility in the underserved areas.

A trivial strategy to do so would be to place more stops and lines in underserved areas. However, this may not be the most efficient way to increase accessibility there. Indeed, the ability to reach PoIs might be increased even more by improving PT network close to other nodes, possibly far away, that may enable convenient changes with other important

[a] https://orcid.org/0009-0007-8567-4707
[b] https://orcid.org/0000-0002-5448-6646

Wang, D., Araldo, A. and Chau, M.
Public Transport Network Design for Equality of Accessibility via Message Passing Neural Networks and Reinforcement Learning.
DOI: 10.5220/0013166000003890
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 3*, pages 619-630
ISBN: 978-989-758-737-5; ISSN: 2184-433X

lines. In general the PT network extends the interdependencies between locations and PoIs far beyond those that are in physical proximity. Therefore, to reduce the inequality of the distribution of accessibility it is always required to take the entire PT graph into consideration, rather than just around the local areas where we want improvement. This makes our problem particularly challenging.

Most cities already have an existing PT network, and the need to build lines from scratch is very limited. Re-designing the whole PT network is also not an option as it would lead to major costs to the operators. For this reason, in this work, we assume a core PT network (e.g., metro) that does not change, and we only tackle the design of some bus lines, which comes at a limited infrastructure cost. By focusing on bus network design only, we aim to achieve important reduction of accessibility inequality with relatively low expenses for the operator.

The contribution of this paper is a novel approach to PT network design, in which the non-trivial interdependencies involved into the accessibility metrics are captured via a Message Passing Neural Network (MPNN) (Wang et al., 2023; Gilmer et al., 2017; Maskey et al., 2022) and a Deep Reinforcement Learning (RL) agent. While MPNN and RL have been used to solve canonical optimization problems on graphs, to the best of our knowledge we are the first to use them for PT network design. To reduce inequality, we propose a simple yet effective approach, consisting in using quantiles of the accessibility metrics as objective function.

Numerical results in a scenario inspired by Montreal show that our method effectively reduces accessibility inequality, more effectively than metaheuristics classically used for PT design. This improvement is due to the capability of the MPNN to capture the structure of the PT network and its relation with the PoIs, while metaheuristics do no learn any dependencies and restrict themselves in randomly exploring the space of designs.

2 RELATED WORK

Transport Network Design Problems (TNDPs), and in particular Public Transport Network Design Problems (PTNDPs) can be at a strategic level or an operational level. At a strategic level, a PT planner aims to decide the route of the different lines as well as their frequencies. At an operational level, a PT operator organizes the service in order to match the decisions taken at the strategic level, deciding precise time tables, as well as crew and vehicle scheduling. In this paper, we focus on PTNDPs at a strategic level. Reviews of strategic-level PTNDPs are provided in (Farahani et al., 2013; Gkiotsalitis, 2022). The methods generally used to solve PTNDPs can be divided into two categories: mathematical programming methods (Section 2.1) and search-based heuristics methods (Section 2.2). We use instead graph-based reinforcement learning (Section 2.3). The latter has been applied to solve several combinatorial problems and has also few applications in Transport. However, it has not been used for PT planning (lines design), with one exception (Yoo et al., 2023). However, (Yoo et al., 2023) only designed the bus network based on some cost function. Their article did not consider the impact of the structure of pre-existing metro lines on the design of new bus lines, nor did it consider optimizing the inequality of accessibility. Their method is difficult to apply to our problem.

2.1 Mathematical Programming Methods

TNDPs usually be formulated as non-linear programming models. Solvers are used to solve these models. To ensure that their models are reasonable, they usually need to set numerous constraints (Wei et al., 2021; Quynh and Thuan, 2018). For a realistic sized problem, it is difficult to find a suitable solution with this method (Chakroborty, 2003). Therefore, instead of considering a real city, they turned to representing it with a regular geometrical pattern, such as the Continuous Approximation method (Calabro et al., 2023). This method can indeed crudely describe any city through some characteristics, but in the end they cannot be applied to any real city. Because real cities are much more complex than abstract regular geometrical shapes.

Some works try to achieve the purpose of dealing with large-scale real cities by adding more constraints, which often limit the solution space. These works are (Gutiérrez-Jarpa et al., 2018), which first proposed to design metro lines in some predefined corridors, and these corridors with higher passenger traffic were chosen by a Greedy generation heuristic step before any optimization, and (Wei et al., 2019), conducted a real-world case study by predefining corridors and solving a bi-objective mixed-integer linear programming model. However, all these works highly rely on an expert guidance to help reduce the potential solution space, in other words, the results vary with different expert guidance. Expert guidance does not ensure that the optimal solution is not eliminated.

2.2 Heuristics Methods

The most commonly used search-based heuristic methods are Simulated Annealing, Tabu Search, and Genetic Algorithm. Their common method is to initialize a PT design and then gradually optimize the PT design by changing it through heuristics. (Schittekat et al., 2013) develops a bi-level metaheuristic to find the optimal solution of school bus routing problem. The upper level repeats greedy randomized adaptive search followed by a variable neighborhood descent n_{max} times and then finds the best solution to bus stops assignment among these n_{max} times of search, while the lower level finds an exact solution to a sub-problem of assigning students to stops by solving a mathematical programming problem. (Owais and Osman, 2018) uses a Genetic Algorithm to generate a bus route network from With only the Origin–Destination matrix and the network structure of an existing transportation network. In the process of using GA to evolve route design, the connectivity of routes is ensured at every stage of the GA. (Kovalyov et al., 2020) uses a Particle Swarm Optimization method to solve the optimal planning problem of replacing traditional Public Transport with electric. (Barceló et al., 2018) emphasizes the application of metaheuristic algorithms in problems of managing city logistics systems. These methods have a common limitation, that is, during the iterative process, some basic characteristics of the graph are retained, such as connectivity, which is not needed in reality. Therefore, they often can only find local optimal solutions.

Considering all of the above, we need to design a generic algorithm which requires fewer constraints. It does not require expert experience, which is the limitation of Mathematical programming method, and it also can no longer retain some unreasonable characteristics of graph, which is the limitation of Search-based heuristics method. Therefore, we chose the RL-based method.

Closer to our work, the work in (Yoo et al., 2023) solved TNDP by directly using RL, and did not extract the information of PT graph. The work (Darwish et al., 2020) used Transformer architecture to produce the nodes and the graph embeddings, and then solved TNDP via RL. But to make their method feasible, they assumed that the network is a connected graph, which is not needed in our paper. The work (Wei et al., 2020) presented a RL-based method to solve the city metro network expansion problem. Our main difference lies in the different methods used to extract the information on PT graph. They used two 1-dimensional convolutional neural networks to calculate the embeddings for the stations. However, the graph structure is complex, we believe only applying 1-dimensional convolutional neural network is insufficient. In this sense, we proposed to use Message Passing Neural Network (MPNN) to extract the information on PT graph.

2.3 Graph-Based Reinforcement Learning Applications

Some works using Graph-based RL to solve other problems are as follows: The work (Barrett et al., 2020) already coupled it successfully to a DQN on traditional combinatorial problem, such as max-cut problem. The work (Duan et al., 2020) first extracted the information of graphs, based on the recurrent neural networks (RNN), and then combined it with RL to solve the Vehicle Routing Problem. The work (Yoon et al., 2021) improved the transferability of the solution to traffic signal control problem by combining Reinforcement learning and MPNN. The work (Köksal Ahmed et al., 2022) proposed an algorithm for the vehicle fleet scheduling problem, by integrating a reinforcement learning approach with a genetic algorithm. The reinforcement learning is used to decide parameters of genetic algorithm.

3 MODEL

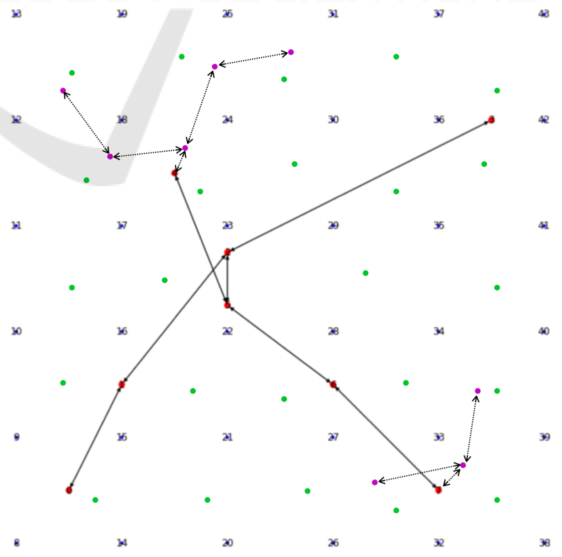

Figure 1: Model of Public Transit*: PT graph $\mathcal{G}$ has 2 metro lines (red points represent metro stations) and 2 bus lines (purple points represent bus stops), in addition, the blue points are the* centroids*, and the green points are the points of interest.*

3.1 Model of Territory and of Public Transport

We partition the study area with a regular tessellation (here we adopt square tiles, but any regular shape can be used), as in Figure 1. The center of each tile is called *centroid* (blue points in Figure 1). The study area also contains Points of Interest (PoIs) often called ”opportunities” in the literature about accessibility. PoIs can be shops, jobs, schools, restaurants, etc. PoIs are depicted as green points in Figure 1.

As illustrated in Figure 1, our model of PT is composed of:

1. Metro lines and metro stations (red points).
2. Bus lines and bus stops (purple points).

Changing metro lines is very costly and time consuming. On the other hand, redesigning bus lines requires much less infrastructure cost and can be done in shorter time. In this paper, we only focus on redesigning bus lines and keep metro lines unchanged.

We model PT as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{L})$, where $\mathcal{V}$ is the set of nodes, composed of centroids and PT stops $\mathcal{V} = \mathcal{C} \bigcup \mathcal{B} \bigcup \overline{\mathcal{B}}$, $\mathcal{E}$ is the set of edges, and $\mathcal{L}$ is the set of lines. Any PT line l (metro line or bus line) is a sequence of PT stops, linked by edges $e \in \mathcal{E}$. Each edge has a weight, which represents the time used by a vehicle to go from a PT stop to another. A PT line l also has a headway t_l, which is the time between two vehicle departures in the same direction. Since we only optimize bus lines, headway t_l of any metro line remains unchanged. For metro lines, headway t_l can be obtained from real data. Instead, since we build bus lines, we need to calculate ourselves headway t_l for any bus line l. Once we decide the sequence of bus stops composing l, we can get the total length of the line d_l, to go from the first to the last stop. We assume number N_l of buses deployed on line l is fixed in advance. In this case, the cost is always the same since the total fleet size (= kN_l) is fixed. Denoting with s_b the bus speed, headway t_l is:

$$t_l = \frac{d_l}{s_b \cdot N_l}. \tag{1}$$

In reality, the headway could also be a bit higher, due to the time spent by the bus at the terminal before starting the next run.

As in Figure 1, we include in $\mathcal{G}$ the set of centroids $\mathcal{C}$ and the set of points of interest $\mathcal{P}$. We also include edges (in the two directions) between any centroid and all PT stops, between any point of interest and all PT stops, and between any centroid and any point of interest. Note that $\mathcal{V}$ is defined as the set of nodes, $\mathcal{C}$ is a subset of $\mathcal{V}$.

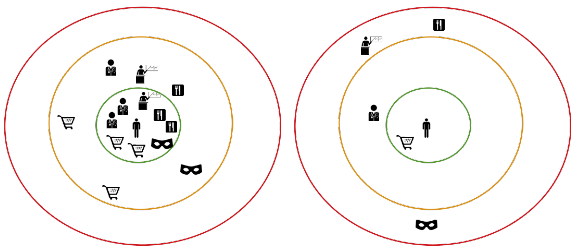

Figure 2: Accessibility example: the location on the left enjoys high accessibility as, departing from it, one can reach many PoIs in little time. On the right, instead, accessibility is poor: few PoIs are reachable and high travel times are required. The left and right locations are typical of city centers and suburbs, respectively.

For a trip from centroid c to point of interest poi, a traveler can choose between different modes of travel. For example, a traveler could simply walk to poi at speed s_w or walk from centroid c to a PT stop (metro station or bus stop), go via PT to another stop, and from there walk to the destination poi. If the stops of two lines are close enough, we will also add the edge between two stops to indicate that passengers can transfer by walking. We consider an average waiting time $t_{l'}/2$ at this station. We assume that travelers always take the shortest path, i.e., the one that allows to arrive at destination with the least time.

3.2 Accessibility

Accessibility measures the ease of reaching PoIs via PT. A simplified depiction is given in Figure 2. Accessibility depends on both land use (which determines where PoIs are) and the transport system (which determines the time to reach each PoI). There are several ways of mathematically defining accessibility. We define the accessibility of centroid c as:

$$acc(c) = \sum_{poi \in \mathcal{P}} \max\left(0, 1 - \frac{T_{c,poi}}{T_{\max}}\right), \tag{2}$$

where $T_{c,poi}$ is the shortest travel time from centroid c to point of interest poi and $T_{\max}$ is a predefined threshold for travel time (e.g., 30 mins). Intuitively, $acc(c)$ measures the number of PoIs that can be reached by individuals departing from centroid c, within time $T_{\max}$. Such PoIs are weighted by the time to reach them, so that the closer a PoI, the more it contributes to accessibility. Our definition is a combination of two classic definitions of accessibility, namely the *isochrone* and *gravity-based* (Miller, 2020). The purely isochrone definition of accessibility has the issue of counting all PoIs the same, despite the difference in travel time to reach time. On the other hand, the purely gravity based definition of accessibility, factors in all PoIs, even those that would require a prohibitive travel time. By combining the two aspects, we solve the aforementioned limitations. Note that accessibility is agnostic to demand: it does not describe where people *currently go* but it measures where they *potentically go* (Miller, 2020). When modifying the PT structure, we modify this "potential of mobility" and thus we also impact where people *will go*. For this reason, mixing current demand with the definition of accessibility would be misleading, as current demand is implicitly invalidated by PT design actions.

We define the global accessibility of graph $\mathcal{G}$ as

$$acc(\mathcal{G}) = \sum_{c \in \mathcal{C}} acc(c). \tag{3}$$

Classic efficiency-based optimization of PT would aim to maximize $acc(\mathcal{G})$. Instead, our aim is to reduce the inequality in the geographical distribution of accessibility, and thus we choose to consider the accessibility of the centroids that suffer from the worst accessibility, as we aim to concentrate improvement in such zones.

One could be tempted to apply max-min optimization, trying to maximize the lowest accessibility in the territory. However, when we tried that, we obtained poor results. Indeed, we were ending up improving areas that were remote and often uninhabited. Often, the improvement was enjoyed by too few locations. We therefore propose to maximize some bottom quantile of the accessibility distribution. To the best of our knowledge, this simple yet effective idea has not been explored so far. We define the following accessibility metric, related to the qth quantile:

$$acc^q(\mathcal{G}) = \sum_{c \in \mathcal{C}^q} acc(c), \tag{4}$$

where $\mathcal{C}^q$ defines the set containing the $q\%$ of centroids with the least accessibility. Note that $acc(\mathcal{G}) = acc^{100}(\mathcal{G})$.

3.3 Problem Definition

Let us consider a PT graph $\mathcal{G}$ and a set $\mathcal{B}$ of n_b *candidate bus stops*. Set $\mathcal{B}$ is contained in set $\mathcal{V}$ of nodes of $\mathcal{G}$. Set $\overline{\mathcal{B}} = \mathcal{V} \setminus \mathcal{C} \setminus \mathcal{B}$ is the set of *non candidate stops*, i.e., the ones that will not be used to create the new lines.

In broad terms, we consider the problem of the PT operator to design k bus lines $\{l_1, \dots, l_k\}$ (where number k is fixed a-priori), passing by these n_b bus stops, in order to reduce inequality of accessibility. Any stop in $\mathcal{B}$ may be already part of pre-existing lines or not. The problem at hand may emerge in case a PT operator wishes to build additional bus lines, passing by stops $\mathcal{B}$. Another case is when a PT operator wishes to redesign current bus lines, while reusing current bus stops.

In quantitative terms, we wish to find graph $\mathcal{G}^*$, which is as $\mathcal{G}$ but also contains additional lines $l_1, \dots, l_k$, such that $acc^q(\mathcal{G})$ is maximized. Specially, if we define:

$$x_{i,j} = \begin{cases} 1, & \text{if bus stop j in line } l_i. \\ 0, & \text{if bus stop j not in line } l_i. \end{cases} \tag{5}$$

We aim to solve the following optimization

$$\max_{\sigma(l_1),\sigma(l_2),\dots,\sigma(l_k)} acc^q(\mathcal{G}), \tag{6}$$

subject to the following constraint:

$$\sum_{i=1}^{k} x_{i,j} = 1, \forall j = 1,2...,n_b. \tag{7}$$

$$\sum_{j=1}^{n_b} x_{i,j} \geq 2, \forall i = 1,2...,k. \tag{8}$$

According to the values of $x_{i,j}$, we can define each line:

$$l_i = \{b_{i_1}, \dots, b_{i_{n_i}}\}, \forall i = 1,2...,k. \tag{9}$$

Calculating accessibility needs to find shortest path in current PT graph, which makes impossible to give specific formulation of accessibility. Therefore, this problem belongs to the integer programming of a black-box function. Constraint 7 ensures that each candidate stop is assigned to a line. With this constraint, we are forbidding a node to be part of multiple lines. However, this limitation can be easily removed by "duplicating" the same real stop into multiple candidate stops in our model. Constraint 8 means that each line has at least two bus stops. In addition to determining which stops each line contains based on the value of $x_{i,j}$ and Constraint 9, the permutation function $\sigma(l_i)$ (in Formulation 6) is also needed to determine the order of the stops in each line l_i.

Note that maximizing the accessibility of the bottom quantiles means, indeed, to improve the accessibility of the poorest locations. Observe that, selecting only the location with the worst accessibility as the objective function generally returned, in our preliminary experiments, not reasonable results, as it concentrates all the optimization effort to just few, possibly very remote, locations, where improving accessibility is anyways hopeless. Our idea of maximizing the bottom quantiles avoid this kind of biased results.

4 RESOLUTION METHOD BASED ON GRAPH REINFORCEMENT LEARNING

We decompose our problem (§3.3) as a bi-level optimization: in the upper level, we partition the candidate bus stops in k subsets. In the lower level, each subset will be transformed in a line, deciding the order of the stops.

4.1 Markov Decision Process Formulation

Let us denote by $\mathcal{G}$ the initial graph, i.e., the one when no new bus lines have yet been added. We model the upper level problem as the following Markov Decision Problem (MDP):

- **States.** A state is a partition $S = (l_1, \ldots, l_k)$ of candidate stops, where $l_i = \{b_{i_1}, \ldots, b_{i_{n^i}}\}$ is the set of bus stops assigned to line l_i.[1] Each bus stop is assigned to a single line. Given any state S, we build the line corresponding set l_i of stops, $i = 1, \ldots, k$. To transform set l_i in a line we need to decide the order in which the stops in l_i will be visited. Such an order is calculated via a heuristic (Section 4.4). Given sate $S = (l_1, \ldots, l_k)$ and having defined the lines corresponding to $l_i, i = 1, \ldots, k$, we add to graph $\mathcal{G}$ the edges corresponding to those lines and obtain a new graph $\mathcal{G}(S)$.
- **Actions.** At each step, our optimization agent shifts a bus stop b_i from its current line l_o to a target line l_t. The action is defined by a tuple $a = (b_i, l_t)$. The state changes from S to S': state S' is equal to S, except for line l_o which becomes $l_o = l_o \setminus \{b_i\}$, and for l_t, which becomes $l_t = l_t \bigcup \{b_i\}$. The action of changing a bus stop to its own line is not admitted. Observe that ours is a Deterministic MDP (Dekel and Hazan, 2013), i.e., arrival state S' can be deterministically calculated from departing state S and action a.
- **Rewards.** The instantaneous reward collected when applying action $a = (b_i, l_t)$ on state S, is
$$r(S,a) = acc^q\left(\mathcal{G}(S')\right) - acc^q\left(\mathcal{G}(S)\right), \quad (10)$$
where parameter q must be chosen in advance.
- **Policy**: During training, our agent follows an ε-greedy policy. At test time, actions are chosen greedily with respect to the Q-values but our agent keeps exploring with a random action every time it finds a local optima.

[1]For simplicity of notation, we use the same symbol l_i to denote a line and also the set of stops assigned to it.

The sizes of the state space and the action space are $k \cdot n_b$, considering a matrix with k lines and n_b bus stops can represent any state and action.

4.2 High-Level View of the Optimization Approach

Due to the high size of the action and state spaces, enumerating all the states and actions and learning a Q-function that takes directly those states and actions as input is hopeless. Therefore, as common in graph-related optimization tasks, we resort to a Message Passing Neural Network (MPNN) (Wang et al., 2023; Gilmer et al., 2017; Hameed and Schwung, 2023; Maskey et al., 2022). Via a MPNN, we embed each node in a low dimension Euclidean space. Such representation captures the "role" of that node within the graph, based on the direct or indirect connections with the other nodes. The process of node embedding is thus able to capture the structure of a graph, so that the RL agent can take decisions that take such structure into account.

A MPNN takes as input PT graph $\mathcal{G}(S)$ of current state S, then it outputs the Q value for each action. Next, the Greedy Policy performs an action according to the Q values. At last, the reward, which is the change of accessibility metric, helps to update parameters of MPNN. The following section will introduce MPNN in more detail. Every time we shift a bus stop from a bus line to another line, a reasonable method of deleting and inserting bus stops in a line is presented in Section 4.4.

4.3 Message Passing Neural Network

Let us associate to each candidate stop b a feature vector x_b. Let us denote with X the matrix of the feature vectors of all stops and with M_{adj} the adjacency matrix, where element (i, j) is 1 if there is a line in which bus stop b_i comes right before b_j, and 0 otherwise.

A MPNN calculates a vector μ_b, called *embedding*, for each candidate bus stop b. Embedding μ_b is a function of feature vector x_b , of feature matrix X, and of adjacency matrix M_{adj}.

In the following, all vectors like Θ_j denote parameters that learned during training. The embedding of bus stop b is initialized as

$$\mu_b^0 = f(x_b, \Theta_1) \in \mathbb{R}^n. \quad (11)$$

The vector of edge embeddings, one per each edge, is initialized as

$$w^0 = g(M_{\text{adj}}, X, \Theta_2) \in \mathbb{R}^m. \quad (12)$$

Let us denote with $\mathcal{N}(b)$ the neighbor bus stops of bus stop b. Note that since the bus lines change from a state S to another, $\mathcal{N}(b)$ may also changes for bus stop b.

The information is then shared to each of the nodes' neighbors through T rounds of messages. The message passed to candidate stop b at round $t+1$ is:

$$m_b^{t+1} = M(\mu_b^t, \{\mu_u^t\}_{u\in\mathcal{N}(b)}, \{w_{ub}^t\}_{u\in\mathcal{N}(b)\}}, \Theta_3^t) \in \mathbb{R}^{n'}, \tag{13}$$

where w_{ub}^t is the embedding of edge between bus stop u and bus stop b after t times iterations.

Embedding u_b^{t+1} of candidate stop b by message:

$$\mu_b^{t+1} = U(\mu_b^t, m_b^{t+1}, \Theta_4^t) \in \mathbb{R}^n, \tag{14}$$

M and U are respectively message and update functions at round t. After T rounds of message passing, a prediction is produced by some Readout function, R. In our case, the prediction is the set of Q-values corresponding to the actions of the changing bus stops and their target line:

$$\{Q(S,a)\}_{a\in\mathcal{A}} = R(\{\mu_b^T\}_{b\in\mathcal{B}}, \Theta_5), \tag{15}$$

where $\mathcal{A}$ is the set of actions, and $\mathcal{B}$ is the set of bus stops. Note that μ_b^T is calculated via Formula 13 and Formula 14, thus varys for different state S. Message function M, Update function U, and Readout function R, as well as the embedding functions f and g, are all neural network layers with learnable weights $\{\Theta_1, \Theta_2, \{\Theta_3^t\}_t, \{\Theta_4^t\}_t, \Theta_5\}$. Every time our agent takes an action and gets a reward r. The One-Step Q-learning loss is:

$$Loss(S,a) = (\gamma \cdot max_{a'} Q(S',a') + r - Q(S',a))^2, \tag{16}$$

where γ is a Discount factor. We update learnable weights via Stochastic Gradient Descent of loss function 16. (Khalil et al., 2017) also used the same loss to deal with the combination problem on the graph. From (Blakely et al., 2021), the total time complexity of the forward step and the backward step of MPNN is $O(T\cdot(|\mathcal{V}|+|\mathcal{E}|))$ for sparse matrix of PT graph with embedding vector dimension unchanged. Combined with action space is $k \cdot n_b$ in Section 4.1, the time cost of each step is linearly related to the sum of the number of nodes and edges in the graph $\mathcal{G}$.

4.4 Sorting Algorithm

Recall that state S is a partition $(l_1,\ldots,l_k)$ of set $\mathcal{B}$ of candidates nodes. State S just establishes to which line each candidate node belongs. However, to transform any set l_i into a line, a certain ordering of its stops must be established.

In this section, we will introduce the method of determining the order of bus stops. This optimal method should also maximize our accessibility objective function. However, since it is a Traveler Salesman Problem, defining such a function seems utterly complex or with a very high computational cost. Considering that order must be determined every time the RL agent selects a new bus stop subset, we decided to use the shortest path algorithm as a proxy for the maximum accessibility path algorithm. We understand that this approach is suboptimal as we optimize distances, but accessibility is a measure completed over the whole graph. Nevertheless, using this algorithm optimizes the headway of our bus lines, and thus renders a good enough accessibility with low computational costs.

4.5 Reinforcement Learning Equality Algorithm

Given the number of bus lines k and the PT network $\mathcal{G}$, we propose a Reinforcement Learning Equality algorithm (Algorithm 1) to plan k bus lines efficiently, in order to optimize the accessibility objective function $acc^q(\cdot)$ ($q = 20, 50, 100.$). Our base idea is to get a better state-action function Q by updating the MPNN network. Note that only when our objective function value is better than before, we will update our MPNN network. We still explore different actions (and thus different graph configurations) for each step, also the ones that decrease the score with respect to the last graph obtained. We just remove from exploration those graphs that are worse than the initial one. If $acc^q(\mathcal{G})$ has not changed in the past certain iterations (e.g., 5 iterations), the episode will be ended. Algorithm 1 terminates when a certain time threshold (e.g., 1h) is exceeded. In fact, the initial graph $\mathcal{G}$ of our algorithm can be different for each episode. Therefore, it has the ability to learn among different PT networks.

5 EVALUATION

5.1 Considered Scenario

To showcase our agent, we consider a simplified version of Montreal. Note that we do not aim to set up a realistic mobility scenario with all the needed details for transport planning. This would indeed require years of effort, by specialized transport consultant companies, and it is out of our scope. Here, our intention is to show that the proposed approach is effective from a methodological and algorithmic point

```
Data: Number of lines $k$, initial graphs $\mathcal{G}$, quantile $q$
Result: The best accessibility $acc^q_{best}$
while Running time threshold is exceeded do
    Initialization: Randomly partition the bus stops between the lines to obtain initial state $S$;
    Sort the lines (§4.4), and update $\mathcal{G}$;
    Set $acc^q_{best} = acc^q(\mathcal{G})$;
    while $acc^q(\mathcal{G})$ has not changed in the past 5 iterations do
        Predict $Q = MPNN(\mathcal{G}, S)$ (15);
        Update the state with action $a = \text{argmax}_a Q(S, a)$;
        Sort each line $l_i$ (§4.4);
        Calculate headway $t_{l_i}$ via (1) for each $l_i$;
        Update graph $\mathcal{G}$ with the new bus lines and headway ;
        Compute $acc^q(\mathcal{G})$ (see (4)) ;
        if $acc^q_{best} > acc^q(\mathcal{G})$ then
            Set $acc^q_{best} = acc^q(\mathcal{G})$;
            Compute loss via (16);
            Update learnable weights $\{\Theta_1, \Theta_2, \{\Theta^t_3\}_t, \{\Theta^t_4\}_t, \Theta_5\}$ of MPNN by Gradient Descent;
        else
            Continue;
        end
    end
end
```

Algorithm 1: Online Reinforcement Learning Equality algorithm.

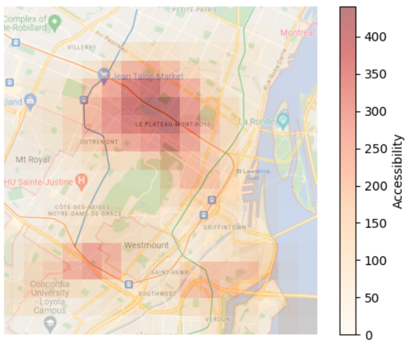

Figure 3: Accessibility of Montreal Metro network.

of view. For this reason, any cost-benefit analysis of our resulting PT network in Montreal, and any detailed environmental assesment is out of scope for this algorithmic focused paper. Observed that one could replace accessibility with cost or km traveles or any other transport related metric. However, we chose to focus on the most difficult metric among those used in common PT design problem, as to optimize accessibility quantiles requires the model to be able to deeply capture the inter-relation between the graph and its surrounding environment, as well as spatial inequalities. This is the focus and the interest of this paper.

From the General Transit Feed Specification (GTFS) data of Montreal ((STM, 2023)), we take the station locations, the sequence of stations of all lines. We assume the PT operators wishes to keep the metro (subway) network as it is, but wishes to build bus lines in order to reduce the inequality of the geographical distribution of accessibility. Therefore, set $\mathcal{B}$ of candidate stops consists of all bus stops, while set $\overline{\mathcal{B}}$ of non-canidate stops are the metro stops. Figure 3 shows the accessibility distribution of accessibility resulting from initial PT graph $\mathcal{G}$ of Montreal, which we assume consists of only the current lines. These assumptions may correspond to the case in which the PT operator wishes to completely redesign bus lines (so that we can remove all bus lines from our initial graph $\mathcal{G}$). The metro network is composed of 4 lines of different sizes. It covers mostly the center of Montreal, which delimits our environment boundaries. In our case, bus stop locations can change over the year. We wanted to check that our method conveys good results, no matter the set of stops. This is the reason that our results are shown for 100 different instances of the problem, each with a different set of stops. The real locations of bus stops in Montreal can be easily added to the training set as an instance. To establish preliminary results, we first limit the number of bus stops to an arbitrary low number of 72. We generate the bus stops as follows. We tessellate the territory with a regular grid. Note that this tessellation is not necessarily the same as the one described in Section 3.1. Within each cell of such a grid a bus stop is created with a random location within the cell. By generating bus stop locations in this way, we ensure uniformity of the bus stops on a larger scale. Consequently, all areas of the territory have access to the public transport network to all regions. We extract points of interest from Open Street Map (OpenStreetMap contributors, 2017) in combination with the Overpass API and assign to the centroids the point of interest number in their area. For the points of interest, we select some of the main amenities in a city (schools, hospitals, police stations, libraries, cinemas, banks, restaurants, and bars). Some of these amenities present a distribution which is relatively uniform over the map (e.g., schools) while some others have higher densities in popular streets (e.g., restaurants). Scenario parameters are in Table 1.

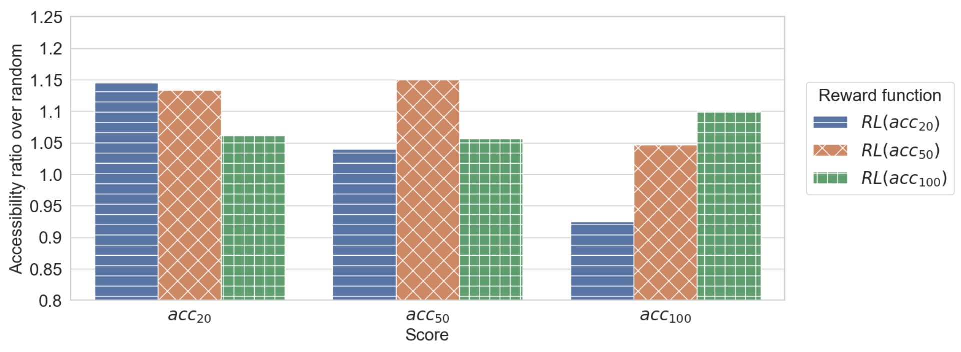

Figure 4: Accessibility ratio ($\frac{acc_a(RL)}{acc_b(random)}$, a,b = 20,50 or 100) via our Reinforcement Learning Equality algorithm (Algorithm 1) against the random search algorithm. x-axis shows different metrics of the Random search algorithm, and the labels show the different metrics used by RL methods.

Table 1: Scenario parameters.

Parameter	Value
Number of bus stops n_{bs}	72
Number of lines k	3
Maximum accessibility time $T_{\max}$	30 minutes
Walking speed s_w (Ali et al., 2018)	4.5 km/h
Bus speed s_b (Ishaq and Cats, 2020)	28km/h
Fleet size per line N_l	10
Distance between adjacent centroids	1km
Discount factor γ	0.95

5.2 Baselines

We compare the performance of our algorithm to two baselines. To compare the algorithms, we define a maximum running time of 1-hour for each method and check which is the best bus lines deployment found. Specially, 1 hour is the training time for our algorithm, our testing time is too short and can be ignored. For Baselines, they keep looking for better graphs within one hour. After one hour, we simply stop searching and output the current optimal graph of Baselines.

5.2.1 Random Search Algorithm

Random states of the form $S = (l_1, \ldots, l_k)$ are generated, each corresponding to a random partition of set $\mathcal{B}$ of candidate stops. At every generated state $S = (l_1, \ldots, l_k)$, each set l_i is sorted as in Section 4.4 to generate the corresponding lines. Then the corresponding graph $\mathcal{G}^{\text{rnd}}$ is constructed, and accessibility $acc^q(\mathcal{G}^{\text{rnd}})$ is computed. This process is repeated until the running time threshold is exceeded. The largest value of $acc^q(\mathcal{G}^{\text{rnd}})$ found at that point is then returned. This random search algorithm has the advantage over our algorithm to visit very diverse states and no time for computing node embeddings, rewards, etc. The number of states visited by this algorithm is much larger than the one visited by our approach, within the same running time threshold.

5.2.2 Genetic Algorithm

The second baseline is a genetic algorithm (Algorithm 2). It is a popular metaheuristic to solve combinatorial problems, and PT design problems in particular (Farahani et al., 2013). In transit network design, state-of-the-art algorithms are generally metaheuristic. There are many flavors of GA for transit network design. Our GA algorithm is an effort to reproduce those. However, due to difficult reproducibility of such methods, we cannot claim our GA algorithm is exactly the same as the one in the state of the art. We are sure that we did our best to improve our GA algorithm, in the effort to have it represent the state of the art. We adopt $acc^q(\mathcal{G})$ as the fitness function. Next, the design of the evolutionary functions must be done carefully so that the new generations inherit good genes from their parents. Therefore, the most important function that must be modified for that case is the crossover function. More popular approaches, like Order Crossover 1 (OX1) preserve relative order of the states and thus their structure, are used in similar problems like TSP (Kora and Yadlapalli, 2017) and VRP (Prins, 2004). Children can benefit from advantageous orderings of the parent nodes, as from our experience, close nodes (sorted by the *SORT* function) are often good to increase accessibility.

5.3 Results

For the first comparison, we train our Reinforcement Learning Equality agent and test it on the same bus stop distribution. The random search baseline run with different accessibility metric. In Figure 4, the different colors correspond to different accessibility metrics adopted as reward of our RL algorithm. Fig-

```
Input: Size $n_{pop}$ of the population, number
  of lines $n_{lines}$, number of parents $n_{par}$ per
  generation, timeout $t_{limit}$ of the benchmark,
  probability $P_{mut}$ of an attribute to mutate ;
Result: The best accessibility over all
        generations $acc^{best}$.
Initialization: Initialize
  $pop_0 = (I_1, \dots, I_{n_{pop}})$ at generation 0 as
  individuals with random partitions of the
  bus stops between the lines. Sort the lines of
  each individual. Compute $acc_0^{best}$ ;
while Running time threshold is exceeded do
    Select $n_{par}$ best $parents =$
      $Tournament(pop, acc_{20})$ ;
    while $pop_i$ is filled with $n_{pop}$ children. do
        $p_1, p_2 = Sample(parents, 2)$ ;
        $c = OX(p_1, p_2)$ ;
        $c = Mutate(c, P_{mut})$ ;
        Sort the lines of each child
          $c = SORT(c)$ ;
        $pop_i.Append(c)$ ;
    end
    Compute $acc_i^{best} = \max(acc^q(pop))$ ;
end
```

Algorithm 2: Genetic algorithm.

ure 4 shows that our approach outperforms the random baseline by 10%-15% when trained and tested on the same accessibility metric (e.g., the agent optimizes acc_{20} with its reward function and is tested on the same metric). Our algorithm also improves the other different accessibility metrics than the one it has been trained by an average ratio of 5% over the baseline. Only the agent trained on acc_{20} performs worse than the random baseline. Because when we optimize PT with equality (acc_{20}) as the optimization goal, we may lose some efficiency (acc_{100}). Planning bus lines need to make a trade-off between equality and efficiency. It should be noted that the balance between equality and efficiency is a "political" choice, which the transport planner should choose, and it is not possible to define the "best" value of the quantile. A practical option for the transport planner could be to obtain different networks, with different quantiles, and then compare them a posteriori to check which one is the most adapted to the scenario at hand. In any case, our method is agnostic to the "political" choice of q%.

Our agent is also able to apply to different bus stop distributions. We assume that each bus stop remains in the same grid cell, but its position is changed for different distributions. Now we select acc^{20} as our accessibility metric for training and testing on a set of 100 PT graphs with different bus stop distributions.

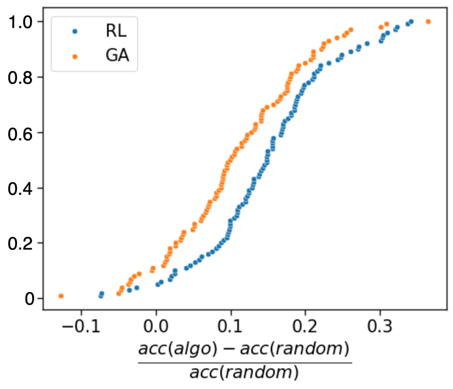

Figure 5: CDF of acc_{20} improvements of different algorithms comparing with random baseline.

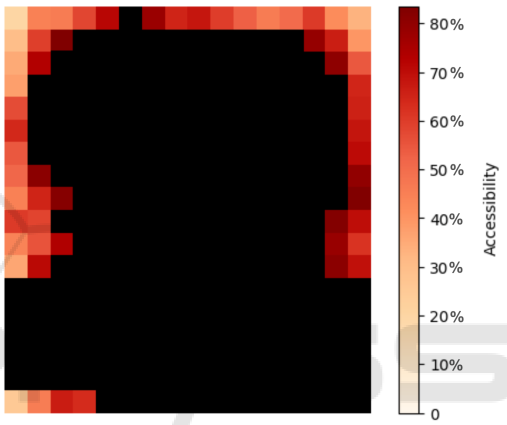

Figure 6: Heatmap of percentage improvement of accessibility (%) via Reinforcement Learning Equality algorithm against the random baseline on underserved areas (The darker red areas indicate that our approach improves more obviously than the random baseline in these areas. Black areas with the best 80% accessibility are not considered).

In Fig. 5, we plot the value of CDF (y-axis) of acc_{20} improvements value

$$R = \frac{acc(algo) - acc(random)}{acc(random)}, \quad (17)$$

where $acc(random)$ is the optimized acc^{20} value by the random search, and $acc(algo)$ is the optimized acc^{20} value by Algorithm 1 or GA. The higher the value of R (x-axis), the better the Algorithm 1 or the GA than the random baseline. We observe in both cases that R follow a normal distribution. Using this data, we conduct a t-test with the null hypothesis (H_0) that $mean(R) = 0$. We run the t-test and get a T-statistic value of 17. From the p-value $p = 3 * 10^{-31}$, we can reject H_0 with confidence more than 99.9%. Thus, we confirm that our approach work better than random baseline on graphs with different bus stop distributions. Similarly, the Genetic Algorithm performs better than the random baseline. Using a similar procedure to compare our Reinforcement Learning Equality algorithm and the Genetic algorithm, we show that our approach performs better on different test graphs with confidence $> 99.9\%$ (the T-statistic value is 8).

We next test the performance of our algorithm on the actual geography of Montreal. Considering acc^{20} as the objective function, so our algorithm focuses more on optimizing those areas with the 20% worst accessibility. We also calculate the value of improvements compared to Random Baseline ($acc^{20}_{\text{RL}} - acc^{20}_{\text{Random}}$) for each centroid with the lowest accessibility in the initial graph. Figure 6 shows that our approach preform better than random baseline in these areas, the improvement in most of these areas exceeded 40%.

6 CONCLUSION

In this paper, we proposed an approach that combined Message Passing Neural Networks (MPNN) and Reinforcement Learning (RL) to optimize bus line design. The objective is to reduce the inequality in the distribution of accessibility provided via Public Transport (PT). Our results showed that MPNN and RL are more effective than commonly used meta-heuristics, as they can capture the PT graph structure and learn the dependencies between lines and the distribution of Points of Interest, where meta-heursitcs restrict themselves to a random exploration of the solution space.

In future work, we will test how our method generalizes to different metro networks, realistically modeled via open data, and how it scales when the problem involves much more bus stops than considered here. In fact, different transit modes can be represented by different graphs. We can connect these graphs to each other to form a multilayer graph. Then, our method can be applied with modifications on the multilayer graph. Of course, for metro networks, once the network infrastructure is put in place, it cannot be easily modified, as with bus lines. Results from the algorithm would thus require a thorough "political evaluation" of the results of the algorithm before implementation. For example, We can set different weights so that the algorithm prefers to modify bus lines rather than metro lines. Moreover, we expect even better results when allowing the same bus stop to be part of multiple bus lines (which is not the case in this work).

The main takeaway of this paper is that combining MPNN and RL is promising to solve PT network design, at a strategic planning phase.

ACKNOWLEDGEMENTS

This work has been supported by the French ANR research project MuTAS (ANR-21-CE22-0025-01) and by Hi!Paris - Paris Artificial Intelligence for Society.

REFERENCES

Ali, M. F. M., Abustan, M. S., Talib, S. H. A., Abustan, I., Abd Rahman, N., and Gotoh, H. (2018). A case study on the walking speed of pedestrian at the bus terminal area. In *E3S Web of Conferences*, volume 34, page 01023. EDP Sciences.

Anable, J. (2005). 'complacent car addicts' or 'aspiring environmentalists'? identifying travel behaviour segments using attitude theory. *Transport policy*, 12(1):65–78.

Barceló, J., Grzybowska, H., and Orozco-Leyva, A. (2018). City logistics.

Barrett, T., Clements, W., Foerster, J., and Lvovsky, A. (2020). Exploratory combinatorial optimization with reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3243–3250.

Blakely, D., Lanchantin, J., and Qi, Y. (2021). Time and space complexity of graph convolutional networks. *Accessed on: Dec*, 31:2021.

Calabro, G., Araldo, A., Oh, S., Seshadri, R., Inturri, G., and Ben-Akiva, M. (2023). Adaptive transit design: Optimizing fixed and demand responsive multimodal transportation via continuous approximation. *Transportation Research Part A: Policy and Practice*, 171:103643.

Chakroborty, P. (2003). Genetic algorithms for optimal urban transit network design. *Computer-Aided Civil and Infrastructure Engineering*, 18(3):184–200.

Darwish, A., Khalil, M., and Badawi, K. (2020). Optimising public bus transit networks using deep reinforcement learning. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7. IEEE.

Dekel, O. and Hazan, E. (2013). Better rates for any adversarial deterministic mdp. In *International Conference on Machine Learning*, pages 675–683. PMLR.

Duan, L., Zhan, Y., Hu, H., Gong, Y., Wei, J., Zhang, X., and Xu, Y. (2020). Efficiently solving the practical vehicle routing problem: A novel joint learning approach. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3054–3063.

EU, E. P. (2019). CO_2 emissions from cars: facts and figures (infographics). https://www.europarl.europa.eu/topics/en/article/20190313STO31218.

Farahani, R. Z., Miandoabchi, E., Szeto, W. Y., and Rashidi, H. (2013). A review of urban transportation network design problems. *European journal of operational research*, 229(2):281–302.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Gkiotsalitis, K. (2022). *Public transport optimization*. Springer.

Gutiérrez-Jarpa, G., Laporte, G., and Marianov, V. (2018). Corridor-based metro network design with travel flow capture. *Computers & Operations Research*, 89:58–67.

Hameed, M. S. A. and Schwung, A. (2023). Graph neural networks-based scheduler for production planning problems using reinforcement learning. *Journal of Manufacturing Systems*, 69:91–102.

Ishaq, R. and Cats, O. (2020). Designing bus rapid transit systems: Lessons on service reliability and operations. *Case Studies on Transport Policy*, 8(3):946–953.

Khalil, E., Dai, H., Zhang, Y., Dilkina, B., and Song, L. (2017). Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30.

Köksal Ahmed, E., Li, Z., Veeravalli, B., and Ren, S. (2022). Reinforcement learning-enabled genetic algorithm for school bus scheduling. *Journal of Intelligent Transportation Systems*, 26(3):269–283.

Kora, P. and Yadlapalli, P. (2017). Crossover operators in genetic algorithms: A review. *International Journal of Computer Applications*, 162(10).

Kovalyov, M. Y., Rozin, B. M., and Guschinsky, N. (2020). Mathematical model and random search algorithm for the optimal planning problem of replacing traditional public transport with electric. *Automation and Remote Control*, 81:803–818.

Maskey, S., Levie, R., Lee, Y., and Kutyniok, G. (2022). Generalization analysis of message passing neural networks on large random graphs. *Advances in neural information processing systems*, 35:4805–4817.

Miller, E. J. (2020). Measuring accessibility: Methods and issues. International Transport Forum Discussion Paper.

OpenStreetMap contributors (2017). Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

Owais, M. and Osman, M. K. (2018). Complete hierarchical multi-objective genetic algorithm for transit network design problem. *Expert Systems with Applications*, 114:143–154.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & operations research*, 31(12):1985–2002.

Quynh, T. D. and Thuan, N. Q. (2018). On optimization problems in urban transport. *Open Problems in Optimization and Data Analysis*, pages 151–170.

Saeidizand et al. (2022). Revisiting car dependency: A worldwide analysis of car travel in global metropolitan areas. *Cities*.

Schittekat, P., Kinable, J., Sörensen, K., Sevaux, M., Spieksma, F., and Springael, J. (2013). A metaheuristic for the school bus routing problem with bus stop selection. *European Journal of Operational Research*, 229(2):518–528.

STM (2023). GTFS data (bus schedules and métro frequency) in Montréal. https://www.stm.info/en/about/developers.

Sun, Y., Schonfeld, P., and Guo, Q. (2018). Optimal extension of rail transit lines. *International Journal of Sustainable Transportation*, 12(10):753–769.

UN (2020). Make cities and human settlements inclusive, safe, resilient and sustainable.

Wang, Z., Di, S., and Chen, L. (2023). A message passing neural network space for better capturing data-dependent receptive fields. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2489–2501.

Wei, M., Liu, T., and Sun, B. (2021). Optimal routing design of feeder transit with stop selection using aggregated cell phone data and open source gis tool. *IEEE Transactions on Intelligent Transportation Systems*, 22(4):2452–2463.

Wei, Y., Jin, J. G., Yang, J., and Lu, L. (2019). Strategic network expansion of urban rapid transit systems: A bi-objective programming model. *Computer-Aided Civil and Infrastructure Engineering*, 34(5):431–443.

Wei, Y., Mao, M., Zhao, X., Zou, J., and An, P. (2020). City metro network expansion with reinforcement learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2646–2656.

Welch, T. F. et al. (2013). A measure of equity for public transit connectivity. *Journal of Transport Geography*.

Yoo, S., Lee, J. B., and Han, H. (2023). A reinforcement learning approach for bus network design and frequency setting optimisation. *Public Transport*, pages 1–32.

Yoon, J., Ahn, K., Park, J., and Yeo, H. (2021). Transferable traffic signal control: Reinforcement learning with graph centric state representation. *Transportation Research Part C: Emerging Technologies*, 130:103321.