Taylor & Francis
Taylor & Francis Group

Applied Artificial Intelligence
An International Journal

ISSN: 0883-9514 (Print) 1087-6545 (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Travel Plans in Public Transit Networks Using Artificial Intelligence Planning Models

Fernando Elizalde-Ramírez, Romeo Sanchez Nigenda, Iris A. Martínez-Salazar & Yasmín Á. Ríos-Solís

To cite this article: Fernando Elizalde-Ramírez, Romeo Sanchez Nigenda, Iris A. Martínez-Salazar & Yasmín Á. Ríos-Solís (2019) Travel Plans in Public Transit Networks Using Artificial Intelligence Planning Models, Applied Artificial Intelligence, 33:5, 440-461, DOI: 10.1080/08839514.2019.1582859

To link to this article: https://doi.org/10.1080/08839514.2019.1582859

APPLIED ARTIFICIAL INTELLIGENCE
2019, VOL. 33, NO. 5, 440–461
https://doi.org/10.1080/08839514.2019.1582859

Taylor & Francis
Taylor & Francis Group

Travel Plans in Public Transit Networks Using Artificial Intelligence Planning Models

Fernando Elizalde-Ramírez, Romeo Sanchez Nigenda, Iris A. Martínez-Salazar, and Yasmín Á. Ríos-Solís

Universidad Autónoma de Nuevo León, San Nicolás de los Garza, México

ABSTRACT

Users of public transit networks require tools that generate travel plans to traverse them. The main issue is that public transit networks are time and space dependent. Travel plans depend on the current location of users and transit units, along with a set of user preferences and time restrictions. In this work, we propose the design and development of artificial intelligence (AI) planning models for engineering travel plans for such networks. The proposed models consider temporal actions, bus locations, and user preferences as constraints, to restrict the set of travel plans generated. Our approach decouples model design from algorithm construction, providing a greater level of flexibility and richness of solutions. We also introduce an integer linear programming formulation, and a fast preprocessing procedure, to evaluate the quality of the solutions returned by the proposed planning models. Experimental results show that AI planning models can efficiently generate close to optimal solutions. Furthermore, our analysis identifies user preferences as the most critical factor that increases solution complexity for planning models.

Introduction

The general road planning problem in a static transportation network with fixed costs is equivalent to the problem of finding the shortest path between an origin and a destination in a directed graph. Each arc in the network has a nonnegative weight that represents the cost function for traversing the arc, which usually correlates to time. The transportation research community has made great strides in the development of road planning algorithms (Bast, Funke, and Matijevic 2006; Sanders and Schultes 2005), which currently can be several orders of magnitude more efficient than Dijkstra's shortest path algorithm (Bauer et al. 2010; Bauer and Delling 2010; Delling et al. 2011; Goldberg, Kaplan, and Werneck 2006; Holzer et al. 2005; Sanders and Schultes 2007).

CONTACT Romeo Sanchez Nigenda romeo.sanchezng@uanl.edu.mx Universidad Autónoma de Nuevo León, San Nicolás de los Garza, NL, México

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/uaai.

However, solution synthesis in public transit networks differs from the general road planning problem because it is time and space dependent. We are interested in finding the earliest arrival time to a destination given the current location of the user, and the transportation units in the network, which can change over time (Pyrga et al. 2008). Furthermore, the optimization problem needs to satisfy individual requirements rather than converging to a single global objective function (Kikuchi 2012). The aim of this article is to propose tools that can flexibly accommodate user preferences and constraints to compute journeys in public transit networks. This is an open area of research, as pointed out by the Transportation Research Board, where Artificial Intelligence models and algorithms could be potentially applied (Kikuchi 2012).

Our work shares this motivation. We do not pretend, at this stage of our work, to compete with domain-specific transportation algorithms yet. Instead, we are interested in applying domain-independent AI algorithms to generate travel plans for public transit networks that consider user preferences. Our analysis sheds light on the current benefits of AI methods for this problem. It is necessary to identify the network properties that increase solution complexity if we want AI models to be applicable to large public transportation networks. To the best of our knowledge, there are very few studies on the subject.

AI Planning is a natural formalism for representing public transit networks. Planning is the process of synthesizing courses of actions that, if executed from a specified initial state (e.g., an initial location in the transportation network), will achieve a set of objectives (e.g., the desired destination) (Ghallab, Nau, and Traverso 2004). Our models use the Planning Domain Definition Language (PDDL – Fox and Long 2003) to encode transit network information. The proposed models consider space information for transportation units in the network (e.g., buses), and consider user preferences (e.g., cost, walking distances) as constraints on the travel plans generated. Given that we use standard AI knowledge representation techniques to generate our models, we can leverage on any domain-independent planning algorithm to compute travel plans. By decoupling model design from algorithm construction, we gain on flexibility and richness of solutions since different paradigms can be used to generate them.

Our empirical evaluation concentrates on assessing the feasibility of the planning models, as well as, the quality of the solutions (i.e., travel plans) generated from them. Moreover, we introduce an integer linear programming model as our baseline to compute optimal journeys on transit networks of different scale, with and without user preferences. We also provide a preprocessing step to prune network size and measure its impact on solution quality. The results show that the planning algorithms can efficiently scale up without compromising solution

quality. Furthermore, our analysis identifies user preferences as the most critical factor that increases solution complexity.

Our results constitute one of the first analysis on the application of AI planning for public transportation networks. The next section introduces related work on the subject. In Section 3, we formally describe the problem that our planning and mathematical models are solving. Section 3.1 presents our planning model representation and Section 4 introduces our integer linear programming baseline. Then, we introduce in Section 5, the pruning techniques implemented to scale up to bigger sized networks. Finally, we present our empirical evaluation and the conclusions of our work.

Related Work

Travel planning in a public transit network greatly differs from the general road planning problem. In a static network with fixed costs, a road planning problem is equivalent to the problem of finding, in a directed graph, the shortest path between two nodes (Dijkstra 1959; Schulz, Wagner, and Weihe 2000). As mentioned earlier, one important difference of public transportation networks with respect to general road networks is that they are time and space dependent. That is, users can travel some parts of the network at specific times, given bus schedules and their locations. Therefore, two of the main challenges in public transit networks are to model the transit timetabling information (Bast et al. 2014), and to define the optimization metrics to traverse such time-space sensitive networks (López and Lozano 2014).

Work by Ishizaki et al. 2010 circumvents the first modeling challenge by calculating the location of buses in the network to update their arrival times at bus stops. Their update procedures are done every minute, but walking distances to bus stops are not considered, which can introduce further delays for the user. Our approach follows the same idea, considering not only bus locations but also walking distances in the planning models. Planning algorithms can then use the available location information to compute better heuristic estimates during their search process.

In addition, there are also approaches for directly modeling timetabling information through *time-expanded* and *time-dependent* graphs (Delling, Pajor, and Wagner 2009; Jariyasunant, Mai, and Sengupta 2011; López and Lozano 2014; Pyrga et al. 2008). In time-expanded models, every event at a location is modeled as a node in a graph. Time-dependent models, on the other hand, consider only one node per location. Although we do not model timetabling information yet, our planning models are suitable for answering queries in real time given the location information from buses and users.

The second challenge, in modeling public transit networks, is the definition of the optimization criteria for traversing the network. The work by Koszelew (2011) in genetic algorithms, optimizes over two criteria: total

travel time (i.e., makespan), and the number of transfers in the network; a criterion also considered by Olczyk and Galuszka (2014). Khoa et al. (2014) generate multi-paths, that is, non-dominated paths between a source and a destination that optimize trip makespan (i.e., travel, walking and transfer time). Multi-modal paths that minimize cost (Huguet et al. 2013), or time (Idri et al. 2017) to drive algorithms are also considered.

In our work, our models support optimization metrics as constraints to restrict the set of travel plans generated. Currently, our models minimize travel makespan, travel cost, and walking distances. However, the flexibility of our models permits the incorporation of additional metrics through numeric PDDL fluents (Fox and Long 2003). This is one major advantage of our proposed models, the flexibility for users to accommodate their preferences, without requiring significant algorithm re-engineering.

To solve the general road planning problem, there are advanced techniques that consider priority queues (Meyer 2001; Thorup 2004), bidirectional search (Dijkstra 1959), labeled arcs (KöHler, MöHring, and Schilling 2005; MöHring et al. 2007), goal-oriented programming (Bauer and Delling 2010), reach-based routing (Goldberg, Kaplan, and Werneck 2006; Gutman 2004), highway-node routing (Holzer, Schulz, and Wagner 2009; Schultes and Sanders 2007), and transit-node routing (2007a; Bast et al. 2007b). Most of these techniques require and exploit the structure of the problem to generate travel plans efficiently. As mentioned earlier, our research decouples model design from algorithm construction. By using standard AI knowledge representation techniques to model transit networks, we can leverage any available domain-independent planning algorithm to compute travel plans.

Route Planning in a Real Public Transit Network Using Artificial Intelligence Planners

For a user that has a set of specific preferences, the AI Public-Routing Planning *AI-PRP* problem consists in traveling from an origin to a destination in a public transportation network, taking into account the location information of the user and the transportation units in the network. We use the PDDL language to encode the transit network information together with the actions of the users to generate solutions for the *AI-PRP* problem.

One of the main benefits of the AI planning languages is that they separate the model of the planning problem in two parts: domain description and the related problem description. Thus, a domain and a problem description form the PDDL-model of a planning problem, and this is the input of a planner software that aims to solve the given planning-problem with an appropriate planning algorithm. The output of the planner is a sequence of actions that aims to a specific route along the network to help users to reach their destinations.

A public transit network PTN is a 5-tuple $PTN = \langle V, E, O, A, W \rangle$ composed by a directed multigraph $G = (V, E)$ where nodes in V correspond to the full set of geographical locations of the network and E corresponds to the set of arcs. Set O compounds the means of transport to traverse an arc $(u, v) \in E$ in the network (e.g., buses from transportation routes or walking). Therefore, each action $x_{iuv} \in O$ represents a transportation unit i traversing the arc (u, v) , where $i > 1$. We set $i = 1$ to represent the user *walking* of an arc. Set A represents user actions a_{1iu} and a_{0iu} for *getting on and off* respectively from transportation units i at geographical locations $u \in V$ of the network.

Finally, W is the set of metric weights for each transportation action $x_{iuv} \in O$ and user action a_{jiu} in the network. For the *AI-PRP* problem, the duration of traversing arc (u, v) with transportation unit i is t_{iuv} , and for walking the arc is t_{1uv} . The time taken for boarding and getting off of a transportation unit i at location u is given by ta_{1iu} and ta_{0iu} respectively. To keep track of the walking distance of an arc (u, v) we use d_{uv} , while the cost of boarding unit i at node u is c_{1iu} .

The planning problem *AI-PRP* of finding a travel plan for a user in a public transit network PTN can be characterized by the 4-tuple $\langle PTN, C, s, g \rangle$, where s and g are, respectively, the initial geographical location and the destination goal of the user (s and $g \in V$). C is the set of preference bounds provided by the user. For example, if $C = \{C_{cost}, C_{walk}, C_{time}\}$, a user can select C_{cost} to restrict the maximum travel cost willing to spend, C_{walk} to set the maximum walking distance willing to engage, and C_{time} to limit the maximum travel time for the whole trip.

The power of AI planning models resides on their capacity to determine the possible transportation actions $x_{iuv} \in O$ that can be coupled with the user transfers $a_{1iu} | a_{0iu} \in A$ to efficiently traverse the transit network PTN from s to g such that user preferences are satisfied. Therefore, a solution to an instance of *AI-PRP* is a travel $\langle PLAN \rangle = \left\{ a_{1is}, x_{isu}, a_{0iu}, \ldots, x_{i'uv'}, \ldots, x_{i''v''g} \right\}$ with the following objective function

$$AI - PRP \langle PTN, C, s, g \rangle = \min \sum_{i \in O} \sum_{(u,v) \in G} x_{iuv}(t_{iuv}) + \sum_{(i,u) \in A} a_{1iu}(ta_{1iu}) \quad (1)$$

$$\text{s.t.} \sum_{(u,v) \in G} x_{1uv} d_{uv} \leq C_{walk}$$

$$\sum_{(i,u) \in A} a_{1iu} c_{1iu} \leq C_{cost}$$

The cost weights associated with the plan should be less or equal than the set of preferences over those costs provided by the user. In other words, a plan is valid if their associated costs are less than the cost preferences specified by

the user. Notice that with our representation we can easily modify the objective function to minimize travel distances or costs. However, for the purposes of this paper, we evaluate our models with Equation 1, the minimization of total travel time subject to user preferences.

AI Planning Model

PDDL is an action-centered language, thus in Table 1 we summarize the actions that make up our planning models. The first column of the table describes the five main actions of the *AI-PRP* problem. The second column contains the parameters that are considered to execute the related action. The third column shows the applicability conditions of the actions, that is, the conditions that must hold in the world for an action to be applicable. The fourth column shows the intended effects of the actions. Finally, the last column indicates the cost functions involved by the actions.

For example, *walk* takes as input parameters the passenger and two locations u and v , where $(u, v) \in E$. In order for *walk* to be applicable, passenger must be at location u , and the cumulative distance traveled must be less than the passenger's walking preference. At the end of *walking*, passenger is at location v , increasing the total travel time, as well as the total walking distance traveled. Action *get on* increases total travel cost and the number of transfers, while *get off* increases travel time. Notice that *move bus without passenger* allows buses to move along their bus lines without any cost.

Figure 1 shows the action transition diagram (ATD) for the *AI-PRP* model actions from Table 1. It describes the actions in the model, as well as their intended effects. Each executed action must account for the user preferences

Table 1. Actions for the *AI-PRP* model.

Action	Parameters	Conditions	Effects	Functions
Walk	passenger location u location v	passenger at u updated walking distance $< C_{walk}$	passenger at v	increase total travel time increase walking distance
Get on	passenger bus location u	passenger at u bus at u updated total cost $< C_{cost}$	passenger on bus	increase total travel time increase total cost increase number of transfers
Move bus with passenger	passenger bus location u location v	passenger in bus bus at u	bus at v	increase total travel time increase total cost increase number of transfers
Move bus without passenger	bus location u location v	bus at u	bus at v	
Get off	passenger bus location u	passenger in bus bus at u	passenger at u	increase total travel time

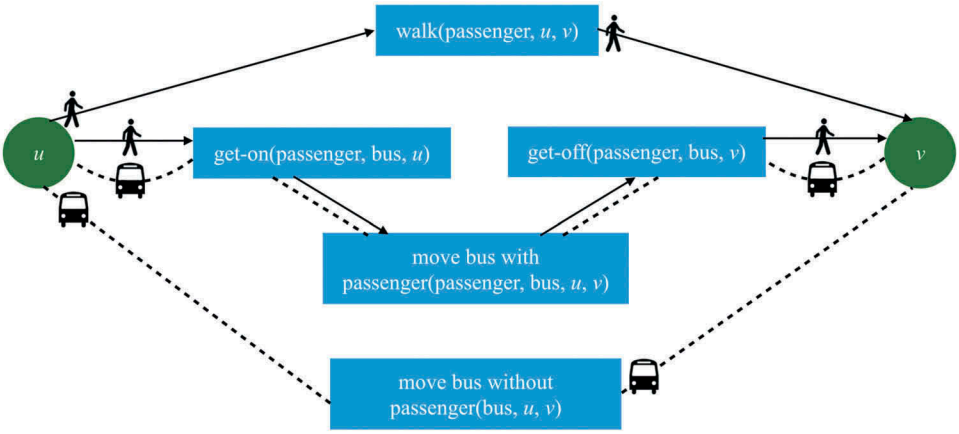

Figure 1. ADT for *AI-PRP* model with applicability conditions and intended effects.

that restrict the set of plans generated by the algorithms. The dotted lines represent the transitions that correspond to transportation actions from O, while the other transitions correspond to user actions from A.

Algorithm 1 shows the PDDL template for the *walking* action, which is a direct implementation of the action description from Table 1 and the ATD from Figure 1. The PDDL action takes as input three parameters, the passenger, the location of origin u, and the destination v. The duration of the walking action t_{1uv} is specified at line two using PDDL numeric fluents, which correspond to metric weights in our problem definition. The action applicability conditions, from line three, are: the user must be at u, there should be a connection in the network from u to v, and the total walking distance, considering the distance from u to v, should be less than the preferred amount of walking defined by the user. The intended effects of the action establish that as soon as the user starts walking he is no longer at the origin, arriving to the destination as an ending effect of the action. Finally, the total travel time and the distance walked by the user get updated.

Algorithm 1 durative-action *walking*

1: Parameters: passenger, locations u and v

2: Duration: t_{1uv}, walking time from u to v

3: Conditions at node u:

- passenger at u and
- arc (u, v) exists and
- walking distance + $d_{uv} < C_{walk}$

4: Effects:

- passenger not in u and
- passenger in v and
- increase walking distance by d_{uv} and
- increase total time by t_{1uv}

Another example is the $Get-on$ action schemed by Algorithm 2. This time, we require that both, the user and the transportation unit (bus), are at the same location u . Notice that we keep track of the total cost of the trip and use it to determine if the cost of taking the transportation unit (i.e., C_{1iu}) will exceed the maximum cost set by the user. The intended effects of the action are that the user is no longer at u and the passenger is inside the transportation unit. At the end, the cost of using the public transport increases the total cost of the travel plan. Notice that we are assuming that the total cost of the trip increases by the number of transfers, a valid assumption in public transportation networks in Latin America. We realize that more complex concession fares could exist in other parts of the world, which might require changes in our methodology.

Algorithm 2 durative-action *get-on*

1: Parameters: passenger, bus i , location u

2: Duration: ta_{1iu} , bus boarding time

3: Conditions at node u :

- passenger at u and
- bus i at u and
- total cost + $c_{1iu} \leq C_{cost}$

4: Effect:

- passenger not at u and
- passenger in bus i and
- increase total cost by c_{1iu}

Notice that the PDDL model description is compact since it concentrates on describing how and when a *lifted operator* is valid to instantiate. For example, the *get-on* operator is lifted because it requires input parameters corresponding to the transportation unit and the location where the action takes place. During model design, we do not need to care about all valid

combinations of these parameters. Decoupling model design from algorithm construction and execution is an advantage of AI planners.

Mathematical Programming Model

To evaluate the performance of AI planning algorithms, an integer linear programming formulation (MILP) is proposed. Nevertheless, to be able to build the MILP, we have relaxed the identification of transportation units and their locations. We consider only route numbers to relate network segments to public transportation. This mathematical model, under this relaxation, generates optimal solutions to the travel plan generation in a public transportation network which allows us to compare the efficiency of the *AI-PRP* models.

Recall from Section 3 that $G = (V, E)$ is a multigraph, O conforms the means of transport, and A represents user transfers. Under these considerations, the following notation is used in the mathematical model:

Variables

The first set of binary variables of the MILP correspond to the means of transport for traversing the network, where $u, v \in V$:

$$x_{iuv} = \begin{cases} 1 & \text{if the user moves from } u \text{ to } v \text{ using conveyance } i, \\ 0 & \text{otherwise.} \end{cases}$$

Recall that we use index $i = 1$ to represent walking actions, and $i > 1$ represents the route number. Furthermore, the following binary variables account for the transfers which will incur in an increase of the cost:

$$r_{iuv} = \begin{cases} 1 & \text{if user changes to transportation } i \text{ to travel from } u \text{ to } v, \\ 0 & \text{otherwise.} \end{cases}$$

The MILP is as follows.

$$min \sum_{i \in O} \sum_{(u,v) \in G} x_{iuv} t_{iuv} + r_{iuv}(ta_{1iu} + ta_{0iv}) \tag{2}$$

$$\sum_{i \in O} \sum_{v \in V|(s,v) \in E} x_{isv} = 1 \tag{3}$$

$$\sum_{i \in O} \sum_{u \in V|(u,g) \in E} x_{iug} = 1 \tag{4}$$

$$\sum_{i \in O} \sum_{j \in V|(j,v) \in E} x_{ijv} - \sum_{i \in O} \sum_{k \in V|(v,k) \in E} x_{ivk} = 0 \quad v \in V \backslash \{s, g\} \tag{5}$$

$$\sum_{(u,v)\in E} x_{1uv} d_{uv} \leq C_{walk} \tag{6}$$

$$x_{iuv} - \sum_{k\in V|(k,u)\in E} x_{iku} \leq r_{iuv} \quad i \in O, u, v \in V \tag{7}$$

$$\sum_{i\in O\setminus 1} \sum_{(u,v)\in E} r_{iuv} c_{1iu} \leq C_{cost} \tag{8}$$

$$\sum_{i\in O} x_{iuv} + x_{ivu} \leq 1 \qquad (u, v) \in E \tag{9}$$

$$x_{iuv}, r_{iuv} \in \{0, 1\} \qquad (u, v) \in E, i \in O$$

The mathematical model's objective function (2) corresponds to the minimization of the total travel time which includes the total walking time, the total travel time in transportation units, and the time required for boarding and alighting the transportation units (i.e., parameters ta_1 and ta_0 from the problem description). Constraints (3) and (4) indicate the origin and the destination of the user. Constraints (5) guarantee flow balance along the network. Constraint (6) restricts the amount of walking time during the trip. Constraints (7) activate variable $r_{iuv} = 1$ when there is a transfer between two lines, while constraint (8) bounds the total cost of the trip. We compute the number of transfers by identifying changes in the means of transportation. As it is shown in Figure 2, transfer and boarding occur on arcs x_{iuv} when index i updates its value between two adjacent locations (e.g., from x_{234} to x_{345}). If i changes its value from one to a value greater than one then a boarding has been detected (e.g., from x_{123} to x_{234}). Getting back the value of i to one represents a user getting off a transport to initiate walking (e.g., from x_{356} to x_{167}). To avoid cycles we introduce restrictions (9). Finally, last equations reflect the nature of the variables.

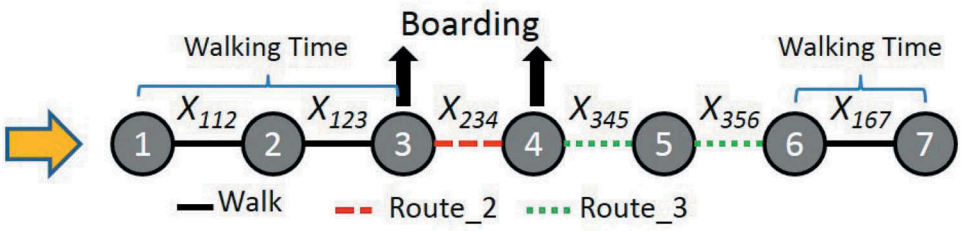

Figure 2. Changes in the means of transport during a travel plan.

Meganodes: Preprocessing Public Transit Networks

As mentioned in the introduction, there has been work in developing preprocessing techniques to reduce the size of transportation networks with the objective of scaling up to larger networks without compromising solution quality. Our motivation for using such techniques in our models is twofold: first, to evaluate the performance of the AI planning algorithms and MILP formulation in more constrained networks, and secondly to feedback some intuition from the planning algorithms to the preprocessing phase to improve the reduction techniques on the network.

We base our preprocessing techniques on Pun-Cheng 2012's work, which introduced the concept of *meganodes*. Meganodes preprocess an original transportation network to include only stops, from different routes, that are within a distance radius that can be covered by walking. At the end of the process, we obtain a subgraph consisting of meganodes and their connections (see Figure 3).

The original implementation of meganodes (Pun-Cheng 2012) uses fixed Euclidean distances to determine if a node is inside the radius of consideration. Such distances cannot include nodes that exceed the maximum walking distance allowed given the structure of the network. Instead, we execute a simplified search from the origin and destination that considers only the walking segments of the network to the bus stops, to determine which nodes to include in the meganodes. For example, in Figure 3, four network nodes in addition to the origin have been considered in the initial meganode since all of them satisfy the C_{walk} walking constraint (i.e., the user preference). Only two of such nodes are bus stops. Once we have the initial and final meganodes, we identify the set of transportation routes that cross them. In our example, we have only two routes (i.e., the red and black lines). From this set of routes, we include those from the origin and the destination that intersect. If no connections are found, the walking factor is increased to consider a greater radius. Notice that we might be discarding journeys that involve routes that cross the meganodes but do not intersect, requiring the user to

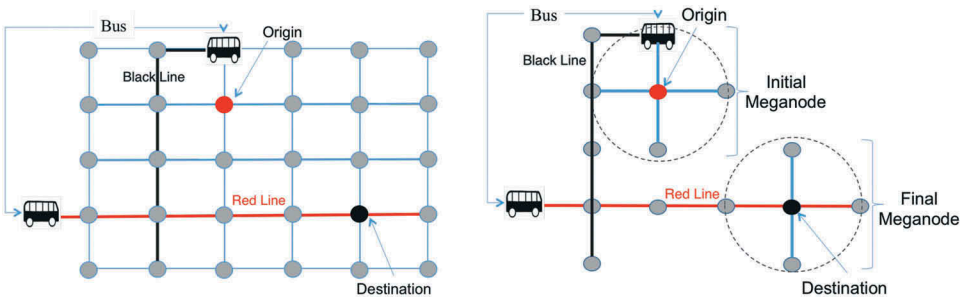

Figure 3. Example of a public transportation network before and after meganode's preprocessing.

walk to perform the transfers. In the next section, we evaluate the proposed models and the impact of the meganodes technique in balancing network size over solution quality.

Results and Discussion

Our case study consisted on public transit networks where the fare scheme is based on ticket purchase per boarding. Such networks are ordinary in Latin America, where a lack of integration between transport systems is common (Varela 2015). We generated randomly two different data-sets to evaluate the proposed *AI-PRP* and the MILP models with the following properties:

(a) Data-set with 120 instances without user preferences, and
(b) Data-set with 120 instances where user preferences are incorporated as constraints on the travel plan.

For data-set b) we considered two types of constraints, one to restrict the total fare of trips, and a second one to limit the amount of walking distance in the network. For the purposes of this evaluation, we minimized total travel time (i.e., the makespan of the travel plan).

After applying our meganodes procedure on these instances, we generated a set of smaller instances as summarized in Table 2. The first column is the number of nodes in the network, the second is the average number of bus stops per bus line, and the third one is the average reduction of nodes after the meganode procedure is applied. We considered six transportation routes for all model instances. The distances between the origin and destination points are at most 75% of the largest Euclidean distance in the network. The amount of route intersections in the network can be up to 50% of the possible combinations.

Although the *AI-PRP* model captures space (i.e., location) information for the transportation units in the network, the current version of the MILP model cannot handle it. The MILP model provides solutions using only routing information. Therefore, to make both models comparable, we assumed transportation units for the planning models at each stop in the network to represent readily available routes.

Table 2. Size of the instances data-sets.

Number of network nodes	Average number of bus stops per route	Meganodes reduction
100	10	40%
144	14	53%
225	22	63%
289	29	65%
400	40	70%
625	78	71%

We selected four planning algorithms for our evaluation that support the properties of our models, and they have been recognized for showing outstanding performance in the International AI Planning Competitions (IPC)[1]: SGPLAN (Chen, Wah, and Hsu 2006), LPG-td (Gerevini, Saetti, and Serina 2003), CPT (Vidal and Geffner 2006), and POPF (Coles et al. 2010). More recent algorithms do not allow numeric fluents in action preconditions and goal conditions when planning with action costs,[2] a PDDL feature needed to represent user preferences as constraints.

SGPLAN is a domain-independent planner that considers subgoal partitioning and conflict resolution for solving problems. SGPLAN was the winner of the suboptimal temporal metric track 2004 IPC and the winner of the satisficing deterministic track in the 2006 IPC. LPG-td is also a domain-independent planner based on stochastic local search and planning graphs that is recommended for domains with numerical quantities and temporal actions. LPG-td won the best-automated planner award in the 2003 IPC and the best performance award in domains involving timed initial literals in 2004. CPT is an optimal constraint-based temporal planner, which combines Partial Order Planning with constraints aimed at problems with high branching factor. CTP won the second prize in the optimal track at the 2004 IPC and distinguished performance in Optimal Planning at IPC 2006. Finally, POPF is a forward-chaining state-based planner that combines linear programming and partial orderings to address temporal-numerical planning problems. POPF won the runner-up award at the 2011 IPC.

To optimally solve our MILP model, we used the General Algebraic Modeling System (GAMS) version 24.2.3 with CPLEX 12.2. GAMS is a high-level modeling system for mathematical programming and optimization. All experiments were conducted on a Lenovo Thinkpad W540 with an Intel Core I7 4740 MQ processor, and 32 GB of RAM running Linux. In the next subsection, we present first our analysis for the full network models, and then our results on the reduced networks generated with the meganodes procedure.

Analysis of Full Network Instances

Tables 3 and 4 show the results of our experiments for complete public transit network models without and with user preferences, respectively. Both tables present the number of nodes, the MILP solution and then the four different algorithms applied to the *AI-PRP* problem. For each resolution method, we present the average processing time for computing the solution, and the average makespan (i.e., total travel time). The last row indicates the percentage of instances for which a solution could be found by the different methodologies. LPG-TD is a stochastic local search planner, therefore, we executed it ten times for each instance and keep its best value in terms of cost.

Table 3. Average running time (in seconds) for finding a solution, and average solution cost (makespan) for full networks without user preferences.

	MILP		CPT		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost	Time	Cost
100	14.27	**406.90**	1.16	412.85	11.43	559.55	0.05	1112.40	**0.13**	406.92
144	47.61	**424.15**	4.82	434.15	14.01	573.10	**0.12**	1350.00	0.25	424.63
225	–	–	30.32	522.40	28.92	611.20	**0.42**	1677.60	0.68	**516.33**
289	–	–	104.80	506.16	23.35	660.55	**0.97**	1879.20	1.31	**503.49**
400	–	–	490.92	587.38	165.07	667.25	**2.90**	2221.20	2.91	**578.35**
625	–	–	–	–	296.75	901.85	29.86	2790.00	**14.26**	**707.01**
% solved instances	33%		73%		100%		100%		100%	

Table 4. Average running time (in seconds) for finding a solution, and average solution cost (total travel time) for complete instances with user preferences on fare and maximum walking distance.

	MILP		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost
100	16.63	**418.80**	71.81	554.00	5.18	841.94	**0.28**	452.62
144	50.99	**433.10**	330.77	574.40	17.07	838.38	**0.68**	453.33
225	–	–	1039.43	636.93	37.97	930.75	**2.46**	556.38
289	–	–	2274.38	725.56	64.87	942.50	**6.01**	536.04
400	–	–	1614.07	691.67	34.21	942.00	**18.14**	620.40
625	–	–	1834.44	1169.00	**47.70**	1419.00	117.77	637.40
% solved instances	33%		57%		40%		100%	

Table 3 shows that, for instances without preferences, CPLEX returned optimal solutions only for the smaller MILP models. On the contrary, three out of four planning algorithms could solve all the evaluation instances. Although CPT, which searches also for optimal temporal plans, can solve only 73% of the instances, performs much better than the MILP methodology in finding optimal solutions. Notice that CPT shows some higher solutions costs than POPF, but this is because of the percentage of problems the algorithm solved.

As expected, average running time is better for the greedy planning algorithms (i.e., LPG-td, SGPLAN and POPF). SGPLAN running times are often the best ones but the quality of its solutions are the worst. The strength of the SGPLAN algorithm comes from partitioning the goal space. However, in our problems, there is only one destination (i.e., one objective) which might affect SGPLAN strategy. Only POPF is competitive with respect to quality (i.e., cost) of the solutions returned, and running time, offering the best trade-off between all the algorithms evaluated. We can conclude, based on the results, that out-of -the-box AI planning problem-solving techniques provide mixed support for finding travel plans in public transit networks. Although some of the algorithms produced running times that might not be practical for real-life applications, POPF's performance constitutes a promising step towards the types of planning algorithms suitable for public transportation networks.

Table 4 presents the results of the data-set that incorporates user preferences as constraints in the models. Specifically, we restricted the fare users are willing to pay for a full journey, and their maximum walking distance during the trip. Notice that despite we are using the same full network models, adding preferences as constraints yields to a different and more complex planning problem. In the taxonomy of planning problems, planning with no preferences corresponds to the general planning problem of *PLAN-EXISTENCE* (Ghallab, Nau, and Traverso 2004), that is, given a problem P one must determine if it has a solution (i.e., a plan). On the other hand, once we add user preferences, we are restricting the set of generated plans, looking for solutions that respect boundary conditions. This scenario corresponds to the problem of *PLAN-LENGTH*, that is, given a problem P determine if it has a solution of $length \leq k$, where *length* and k correspond to user-defined metrics or preferences. It has been shown that *PLAN-LENGTH* is in general harder than *PLAN-EXISTENCE* (Ghallab, Nau, and Traverso 2004) which can be seen in the increase in average running time in our experimentation. The CPT algorithm is not fitted to solve problems with preference restrictions. Only POPF can solve all the instances, increasing up to 725.87% its average running time in the worst case with respect to the non-restricted version. Given that optimal approaches do not scale up, we are not able to assess the quality of the solutions returned by planning algorithms in models with preferences. The next section presents models reduced by the meganodes technique to evaluate the quality factor.

Analysis of Reduced Network Instances with the Meganodes Techniques

Tables 5 and 6 are like the precedent ones but now the instances have been preprocessed with the meganode procedure presented in Section 5. Table 5 shows the results of our experiments for reduced public transit network instances without user preferences while Table 6 corresponds to the reduced instances with user preferences. We want to be able to generate solutions in reduced network models more efficiently without compromising the solution

Table 5. Average running time (in seconds), and average solution cost (makespan) for meganodes models without user preferences.

	MILP		CPT		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost	Time	Cost
100	1.62	**518.25**	167.91	563.67	0.95	535.30	**0.02**	596.95	0.05	518.27
144	2.05	**539.05**	0.38	563.29	0.79	540.70	**0.02**	636.00	0.08	539.07
225	4.16	**589.75**	3.44	557.31	2.13	613.05	**0.07**	765.30	0.19	589.78
289	7.14	**634.05**	10.22	640.00	2.03	650.00	**0.16**	863.57	0.36	634.09
400	11.77	**686.30**	76.06	669.60	8.60	701.15	**0.15**	748.00	0.71	686.35
625	39.77	**776.35**	408.68	859.00	24.10	837.85	–	–	**3.49**	776.42
% solved instances	100%		54%		100%		50%		100%	

Table 6. Average running time (in seconds), and average solution cost (makespan) for meganodes instances with user preferences.

	MILP		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost
100	1.79	**518.25**	1.75	550.40	**0.02**	596.95	0.09	518.27
144	2.24	**539.05**	9.22	551.60	**0.03**	636.00	0.16	539.07
225	4.33	**590.45**	107.94	624.75	**0.10**	765.30	0.53	590.48
289	7.65	**634.05**	52.11	670.15	**0.21**	855.29	1.07	634.09
400	12.64	**687.25**	273.69	714.70	**0.19**	748.00	3.02	687.30
625	45.32	**783.85**	553.82	810.10	–	–	**27.60**	792.07
% solved instances	100%		100%		50%		100%	

quality obtained in full networks. Therefore, our evaluation in this section is aimed at 1) reviewing the models' performance on reduced networks, and 2) assessing the impact of meganodes on solution cost.

Table 5 shows that CPLEX solved all the MILP models to optimality, which in consequence allowed us to evaluate the performance of planning algorithms comprehensively. Notice that MILP models are competitive with respect to the planning models, since they return optimal solutions in high-quality average running times compared to CPT. SGPLAN has the best average times but low-quality solutions. Although POPF is a greedy algorithm, it returned the optimal solutions to most of the evaluation instances at a fraction of the time taken by the other approaches.

Interestingly, reducing the network size is not beneficial to every planning approach. SGPLAN and CPT solved 50% and 26% fewer instances respectively than using the full network counterparts. Our intuition is that reduced networks could be over constraining the original instance, thus some of the greedy planning approaches have problems finding a solution.

We performed the same analysis for reduced network models with user preferences (see Table 6). The MILP solver and POPF again provided the best trade-off between performance and solution cost. Although SGPLAN is the fastest algorithm, it cannot scale up. Again, we can see that the restricted instances are more time consuming than the instances without the user preferences.

To assess the impact of the meganodes technique on the solution cost we observe the percentages changes of the average running time, cost and number of instances solved between full network instances, and the reduced ones obtained after applying the meganodes technique. Table 7 shows such results for the instances without user preferences while Table 8 shows the results for the instances with user preferences.

Notice that the MILP model has an increment of 203% of solved instances, CPT and SGPLAN have decrements of 26% and 50% respectively, while LPG-TD and POPF solved again all the instances thus no changes in percentages. Given that we are showing average running time and average

Table 7. Percentage of change between the results of complete network models from Table 3, and reduced meganodes instances from Table 5.

	MILP		CPT		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost	Time	Cost
Percentage Change										
100	−88.6%	27.36%	14375%	36.53%	−91.68%	−4.23%	−60%	−46.33%	−61.53%	27.36%
144	−95.6	27.13%	−92.11%	29.74%	−94.36%	−5.75%	−83.33%	−52.88%	−68%	26.95%
225	−100%	−100%	−88.65%	6.68%	−92.63%	0.32%	−83.33%	−54.36%	−72.05%	14.22%
289	−100%	−100%	−90.24%	26.44%	−91.30%	−1.51%	−83.50%	−54.07%	−72.51%	25.93%
400	−100%	−100%	−84.50%	13.99%	−94.79%	5.08%	−94.82%	−66.32%	−75.60%	18.67%
625	−100%	−100%	−100%	−100%	−91.87%	−7.09%	∞	∞	−75.52%	9.81%
% solved instances	203%		−26%		0%		−50%		0%	

Table 8. Percentage of change between the results of complete network instances with user preferences with respect to the reduced meganodes instances with user preferences.

	MILP		LPG-TD		SGPLAN		POPF	
Nodes	Time	Cost	Time	Cost	Time	Cost	Time	Cost
Percentage Change								
100	−89.23%	24.04%	−97.56%	−0.64%	−99.61%	−29.09%	−67.85%	14.50%
144	−95.60%	24.46%	−97.21%	−4.17%	−99.82%	−24.13%	−76.47%	18.91%
225	−100%	−100%	−89.61%	−1.91%	−99.73%	−17.77%	−78.45%	6.12%
289	−100%	−100%	−91.30%	−7.63%	−99.67%	−9.25%	−82.19%	18.29%
400	−100%	−100%	−94.79%	3.32%	−99.44%	−20.59%	−83.35%	10.78%
625	−100%	−100%	−69.95%	−30.70%	∞	∞	−76.56%	24.26%
% solved instances	203%		75%		25%		0%	

solution cost, an increase in the percentage change signifies worse performance. For instance, SGPLAN and LPG-TD show improved performance, both in running time and solution cost with respect to the full network instances. The percentage change is more significant in the SGPLAN performance. This is an indication that the meganodes technique does help in improving the performance of greedy planning algorithms. However, this is not always the case. Notice, that POPF decreases the quality of its solution costs (up to 27% on the worst average case).

Recall, from the previous subsection, that POPF returns close to optimal solutions; in consequence, the margin of improvement is minimal. Since our reduction technique is greedy, it considers only routes that intersect with the meganodes given the *walking* radius. Solutions that involve transfers intertwined with walking might get discarded in the reduced networks, leading to sub-optimal performance. This is an open area of improvement for our meganodes technique, that is, find the right balance between network size and solution quality, such that the technique can help algorithms with different levels of performance.

Table 8 shows the percentage of change between values from complete instances with respect to the meganodes ones in the presence of user preferences. Notice that the MILP model, LPG-TD, and SGPLAN increased the percentage of problems solved. Again, there are significant savings in running time for all the algorithms, but mixed results for solution costs in high-performance algorithms. Only the lowest performance algorithms improved their solution costs. Therefore, we can conclude that reducing the size of the transit network is an important performance factor for all the algorithms, which needs to be carefully crafted to avoid losing solution quality in high-performance algorithms.

Conclusions

In this work, we presented two different models to generate travel plans in public transit networks. One is based on AI planning and it uses the PDDL representation

to model different properties from public transportation domains. The second model is based on integer mathematical programming. We consider temporal actions, bus locations, costs, walking distances, and user preferences to generate travel plans. Our main motivation is to provide a generic formalism that can be used to represent public transit network information and users' needs. By encoding our models using standard AI knowledge representation techniques, we can leverage available planning algorithms, and knowledge engineering tools to compute travel plans.

Our empirical evaluation shows that only greedy AI planning algorithms can scale up when complete transit network instances are considered. However, when user preferences are considered, their performances decrease significantly. Only POPF, one of the tested algorithms, can maintain a consistent performance without affecting solution quality. We can conclude, based on the empirical evaluation, that out-of-the-box AI planning problem-solving techniques provide mixed support for finding travel plans in public transit networks.

In addition, we also apply the meganodes reduction technique to our models to produce smaller sized transit networks. The application of meganodes allowed us to solve the mathematical programming models to compute optimal travel plans across all evaluation instances. Furthermore, we verify that POPF, the best planning algorithm in our evaluation, matched optimal solution quality at a fraction of the time taken by the mathematical programming approach. POPF's performance constitutes a promising direction towards the types of planning algorithms suitable for public transportation networks.

We also found out that reduction techniques, although critical for improving algorithms' efficiency, had a negative impact up to 27% on the solution quality of best performers. This is an open area of research that needs to be addressed to provide a better trade-off between performance and the quality of solution synthesis. We are currently deriving knowledge, from planning solutions, to improve the meganodes technique to close the gap between performance and solution cost. We also plan to extend the set of queries our models can answer. Specifically, we are considering extensions to our models to incorporate timetabling information. We are planning to model bus schedules in public transit networks as timed initial literals and external planning events.

The proposed models constitute a step towards alternative, more flexible, and richer paradigms that decouple model design from algorithm construction, which consider user preferences and that leverage on available algorithms, to compute travel plans efficiently for public transit networks.

Notes

1. See http://www.icaps-conference.org/index.php/Main/Competitions.
2. See http://ipc.informatik.uni-freiburg.de/PddlActionCosts for the official IPC documentation.

Funding

This work was supported by the National Council of Science and Technology (CONACyT) of México under grant number [CB-2013/220811]; and the UANL-PAICYT grant [CE333-15].

ORCID

Fernando Elizalde-Ramírez http://orcid.org/0000-0002-0699-2835
Romeo Sanchez Nigenda http://orcid.org/0000-0001-7272-3759
Iris A. Martínez-Salazar http://orcid.org/0000-0002-0037-5764
Yasmín Á. Ríos-Solís http://orcid.org/0000-0003-3785-4440

References

Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., and Werneck, R. 2014. *Route planning in transportation networks* MSR-TR-2014-4. Microsoft Research.

Bast, H., Funke, S., Matijevic, D., Sanders, P., and Schultes, D. 2007b. In transit to constant time shortest-path queries in road networks. In *Proceedings of the ninth workshop on Algorithm Engineering and Experiments (ALENEX)*, ed. D. Applegate and G. S. Brodal, 46–59. Society for Industrial/Applied Mathematics. New Orleans, Lousiana, USA.

Bast, H., S. Funke, P. Sanders, and D. Schultes. 2007a. Fast routing in road networks with transit nodes. *Science* 316 (5824):566. doi:10.1126/science.1137521.

Bast, H., S. Funke, and D. Matijevic. 2006. TRANSIT-ultrafast shortest- path queries with linear-time preprocessing. In *9th DIMACS implementation challenge*, ed. C. Demetrescu, A. Goldberg, and D. Johnson. Piscataway, New Jersey, USA.

Bauer, R., Delling, D., Sanders, P., Schieferdecker, D., Shultes, D., and Wagner, D. 2010. Combining hierarchical and goal-directed speed-up techniques for dijkstra's algorithm. *Journal of Experimental Algorithmics* 15:2.3: 2.1–2.32.31.

Bauer, R., and D. Delling. 2010. SHARC: Fast and robust unidirectional routing. *Journal of Experimental Algorithmics* 14 :4:2.4–42.29. doi:10.1145/1498698.1537599.

Chen, Y., B. W. Wah, and C.-W. Hsu. 2006. Temporal planning using subgoal partitioning and resolution in sgplan. *Journal of Artificial Intelligence Research* 26:323–69. doi:10.1613/jair.1918.

Coles, A. J., Coles, A. I., Fox, M., and Long, D. 2010. Forward-chaining partial-order planning. In *Proceedings of the twentieth International Conference on Automated Planning and Scheduling (ICAPS-10)*, ed. Brafman, R., Geffner, H., Hoffmann, J., and Kautz, H., 42–49. Toronto, Canada.

Delling, D., Goldberg, A. V., Pajor, T., and Werneck, R. F. 2011. Customizable route planning. In *Proceedings of the 10th International Conference on Experimental Algorithms*, ed. P. M. Pardalos and S. Rebennack, 376–87. SEA'11. Isbn: 978-3-642-20661-0. Chania, Crete, Greece.

Delling, D., T. Pajor, and D. Wagner. 2009. Engineering time-expanded graphs for faster timetable information. In *Robust and online large-scale optimization: Models and techniques for transportation systems*, 182–206. ed. Springer Berlin Heidelberg. doi:10.1007/978-3-642-05465-5_7.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1 (1):269–71. doi:10.1007/BF01386390.

Fox, M., and D. Long. 2003. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20:61–124. doi:10.1613/jair.1129.

Gerevini, A., A. Saetti, and I. Serina. 2003. Planning through stochastic local search and temporal action graphs in LPG. *Journal of Artificial Intelligence Research (USA)* 20 (1):239–90.

Ghallab, M., D. Nau, and P. Traverso. 2004. *Automated planning: Theory and Practice*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann. Isbn: 978-1-55860-856-6. San Francisco, CA, USA.

Goldberg, A. V., H. Kaplan, and R. F. Werneck. 2006. Reach for A*: Efficient point-to-point shortest path algorithms. Chap. 1 in *2006 proceedings of the eighth workshop on Algorithm Engineering and Experiments (ALENEX)*, 129–43. Society for Industrial/Applied Mathematics (SIAM). doi:10.1137/1.9781611972863.13.

Gutman, R. 2004. Reach-based routing: A new approach to shortest path algorithms optimized for road networks. In *Proceedings 6th workshop on Algorithm Engineering and Experiments (ALENEX)*, ed. L. Arge and G. F. Italiano, 100–11. SIAM. New Orleans, LA, USA.

Holzer, M., Shulz, F., Wagner, D., and Willhalm, T. 2005. Combining speed-up techniques for shortest-path computations. *Journal of Experimental Algorithmics*:10. doi:10.1145/1064546.1180616.

Holzer, M., F. Schulz, and D. Wagner. 2009. Engineering multilevel overlay graphs for shortest-path queries. *Journal of Experimental Algorithmics* 13:5:2.5–5:2.26. doi:10.1145/1412228.1412239.

Huguet, M.-J., D. Kirchler, P. Parent, and R. Wolfler Calvo. 2013. Efficient algorithms for the 2-way multi modal shortest path problem. *Electronic Notes in Discrete Mathematics* 41:431–37. doi:10.1016/j.endm.2013.05.122.

Idri, A., M. Oukarfi, A. Boulmakoul, K. Zeitouni, and A. Masri. 2017. A new time-dependent shortest path algorithm for multimodal transportation network. *Procedia Computer Science* 109:692–97. doi:10.1016/j.procs.2017.05.379.

Ishizaki, Y., Sasama, T., Kawamura, T., and Sugahara, K. 2010. Determining location of bus and path planning considering bus delay". In *SICE Annual Conference 2010, Proceedings of*, 2436–37. Taiwan.

Jariyasunant, J., E. Mai, and R. Sengupta. 2011. Algorithm for finding optimal paths in a public transit network with real-time data. *Transportation Research Record: Journal of the Transportation Research Board* 2256 (1):34–42. doi:10.3141/2256-05.

Khoa, V. D., Pham, T. V., Nguyen, H. T., and Van Hoi, T. 2014. Multi–criteria route planning in bus network. In *Computer information systems and industrial management*, 535–46. ed. Springer.

Kikuchi, S. 2012. Analysis of public transportation planning and operations. In *Artificial intelligence applications to critical transportation issues*. ed. Transportation Research Board. Washington, DC. USA.

Köhler, E., R. H. Möhring, and H. Schilling. 2005. Acceleration of shortest path and constrained shortest path computation. In *Proceedings of the 4th international conference on experimental and efficient algorithms*, ed. S. E. Nikoletseas, 126–38. WEA'05. isbn: 978-3-540-25920-6. doi:10.1007/11427186_13. Santorini Island, Greece.

Koszelew, J. 2011. An evolutionary algorithm for the urban public transportation. In *Proceedings of the third international conference on computational collective intelligence: Technologies and applications - volume part I*, 234–43. ICCCI'11. Gdynia, Poland: Springer-Verlag. Isbn: 978-3-642-23934-2.

López, D., and A. Lozano. 2014. Techniques in multimodal shortest path in public transport systems. *Transportation Research Procedia* 3:886–94. doi:10.1016/j.trpro.2014.10.068.

Meyer, U. 2001. Single-source shortest-paths on arbitrary directed graphs in linear average-case time. In *Proceedings of the twelfth annual ACM-SIAM symposium on discrete algorithms*, 797–806. SODA '01. Society for Industrial/Applied Mathematics. Isbn: 0-89871-490-7. Washington, D.C. USA.

Möhring, R. H., Schilling, H., Schütz, B., Wagner, D., and Willhalm, T. 2007. Partitioning graphs to speedup Dijkstra's algorithm. *Journal of Experimental Algorithmics*:11. doi:10.1145/1187436.1216585.

Olczyk, A., and A. Galuszka. 2014. Finding routes in a public transport network. A case study. In *Methods and Models in Automation and Robotics (MMAR), 2014 19th international conference on*, 800–03. Miedzyzdroje, Poland.

Pun-Cheng, L. S. C. 2012. An interactive web-based public transport enquiry system with real-time optimal route computation. *Intelligent Transportation Systems, IEEE Transactions On* 13 (2):983–88. doi:10.1109/TITS.2011.2181501.

Pyrga, E., Schulz, F., Wagner, D., and Zaroliagis, C. 2008. Efficient models for timetable information in public transportation systems. *Journal of Experimental Algorithmics* 12 : 2.4: 1–2.4:39.

Sanders, P., and D. Schultes. 2005. Highway hierarchies hasten exact shortest path queries. In *Proceedings of the 13th European Symposium on Algorithms (ESA)*, ed. G. S. Brodal and S. Leonardi, 568–97. Heidelberg: Springer.

Sanders, P., and D. Schultes. 2007. Engineering fast route planning algorithms. In *Experimental algorithms*, ed. C. Demetrescu, 23–36. Springer-Verlag Berlin, Heidelberg. Isbn: 978-3-540-72845-0.

Schultes, D., and P. Sanders. 2007. Dynamic highway-node routing. *Experimental algorithmsed.*, C. Demetrescu, 4525:66–79. Lecture Notes in Computer Science. isbn: 978-3-540-72844-3. doi:10.1007/978-3-540-72845-0_6.

Schulz, F., D. Wagner, and K. Weihe. 2000. Dijkstra's algorithm on-line: An empirical case study from public railroad transport. *Journal of Experimental Algorithmics* 5:12. doi:10.1145/351827.384254.

Thorup, M. 2004. Integer priority queues with decrease key in constant time and the single source shortest paths problem. Special Issue on *{STOC}* 2003. *Journal of Computer and System Sciences* 69 (3):330–53. doi:10.1016/j.jcss.2004.04.003.

Varela, S. 2015. *Urban and suburban transport in mexico city: Lessons learned implementing brts lines and suburban railways for the first Time*. Technical Report. International Transport Forum, Organisation for Economic Co-operation and Development.

Vidal, V., and H. Geffner. 2006. Branching and pruning: An optimal temporal {POCL} planner based on constraint programming. *Artificial Intelligence* 170 (3):298–335. doi:10.1016/j.artint.2005.08.004.