A Gaussian-Prioritized Approach for Deploying Additional Route on Existing Mass Transportation with Neural-Network-Based Passenger Flow Inference

Fandel Lin
Institute of Computer and Communication Engineering
National Cheng Kung University
Tainan, Taiwan
q36084028@mail.ncku.edu.tw

Jie-Yu Fang
Institute of Computer and Communication Engineering
National Cheng Kung University
Tainan, Taiwan
grace963214789@gmail.com

Hsun-Ping Hsieh
Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan
hphsieh@mail.ncku.edu.tw

***Abstract*— Multi-criteria path planning is an important combinatorial optimization problem with broad real-world applications. Finding the Pareto-optimal set of paths ideal for all requiring features is time-consuming and unclear to obtain the subset of optimal paths efficiently for multiple origin states in the planning space. Meanwhile, due to the rise of deep learning, hybrid systems of computational intelligence thrive in recent years. When facing non-monotonic data or heuristics derived from pre-trained neural networks, most of the existing methods for the one-to-all path problem fail to find an ideal solution. We employ Gaussian mixture model to propose a target-prioritized searching algorithm called Multi-Source Bidirectional Gaussian-Prioritized Spanning Tree (BiasSpan) in solving this non-monotonic multi-criteria route planning problem given constraints including range, must-visit vertices, and the number of recommended vertices. Experimental results on mass transportation system in Tainan and Chicago cities show that BiasSpan outperforms comparative methods from 7% to 24% and runs in a reasonable time compared to state-of-art route-planning algorithms.**

Keywords— Constrained route planning, Bidirectional spanning tree, Gaussian mixture model (GMM), Non-monotonicity, Deep Neural Network (DNN)

I. INTRODUCTION

Constrained planning is ubiquitous. Many AI-based or automatic planning tasks can be formulated as the constraint-satisfaction problem (CSP) which involves the assignment of values to variables subject to a set of constraints [1]. Among CSP, though learning-based methods are increasingly popular in solving single-criterion optimization problems; however, few works are developed for dealing with multiple-criteria optimization [2].

Multi-criteria constrained path planning is an important combinatorial optimization problem with broad applications, where algorithms with specific or defined heuristic functions are often utilized. A well-known and state-of-art heuristic algorithm is A* algorithm [3]. A* could always find the path with the cheapest cost if the heuristic function is admissible. More specifically, if the heuristic function (cost function) is monotonic in the problem space, the algorithm could be proved to find the optimal solution. Otherwise, approaches are often infeasible once the heuristic function is not admissible. However, solving only monotonic problems is not realistic since non-monotonicity can be observed frequently in our daily life [4].

This work focuses on solving a non-monotonic and multi-criteria constrained route planning problem and is applied to realistically existing public transit systems for deploying additional transportation routes. Where to deploy additional transportation services such as buses or MRT routes in an urban space is a time-consuming task for governments or public transport authority since several criteria such as total passenger-flow (PF), road structures, fuel consumption, and some other human-specified parameters should be considered. Although several traffic planning softwares (e.g. VISUM, EMME) are already developed, authorities are still often required to provide effectiveness evaluations or near-optimal plans in a timely manner when facing the overwhelming number of requests for deploying new stations or routes from the public. Most importantly, the potential PF is not monotonic during the route construction process so that makes it difficult to construct a route with high PF. Casually adding a new passing area into a forming route could lead to decrease the accumulated PF because of the complex geographical environments and the correlation with existing routes. For example, adequate intersections to existing routes bring more PF since passengers can make transferences between routes; however, PF could conversely decrease when the new route keeps growing too similar to an existing route since passengers would rather take the original route directly than to take the new one. To best of our knowledge, most of the state-of-art route-planning algorithms are not applicable in addressing such the issue.

With a pre-trained PF inference model based on Deep Neural Network (DNN) for regression [5], which can accurately infer PF of arbitrary route, we propose Multi-Source **Bi**directional G**a**u**s**sian-Prioritized **Span**ning Tree (**BiasSpan**) for the non-monotonic multi-criteria constrained route planning problem,

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

which can be applied to deploy additional transportation route in existing public transit systems. The key concept is to smooth the non-monotonic function and input features of the inference model. Specifically, given a set of must-visit stations and a range on the map as constraints from traffic management authorities/ users, BiasSpan recommends a route with near-optimally maximized PF (based on the aforementioned inference model) along with minimized route length. The system flow of the proposed framework is illustrated in Fig. 1.

Experimental results on the mass transportation systems in Tainan and Chicago cities show that BiasSpan outperforms comparative methods from 7% to 24% and runs in a reasonable time compared to several route-planning algorithms.

To conclude, we propose a DNN-based best-first searching algorithm to solve non-monotonic multi-criteria constrained route planning problems with optimization criteria:

- Maximizing PF of a route. (refers to income of fixed fare)
- Minimizing the length of a route. (fuel consumption)

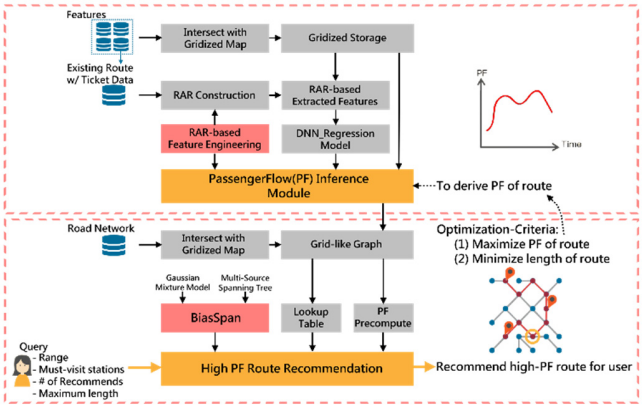

Fig. 1. System Flow for the proposed framework.

II. Background

Research on route planning algorithms in transportation networks has developed over years, where the network is usually modeled as a directed graph in order to utilize Dijkstra's algorithm to compute a best route between two nodes [6]. Several improvements have been made to run Dijkstra's algorithm in almost linear time or using little memory thereafter [7][8]. However, it is too slow for practical applications in real-world transportation networks, which consist of millions of nodes (grids), while instant results are requested [9]. Therefore, to speed up searching an unimodal transportation network, bidirectional search (graph search) [10][11][12][13], goal direction [14][15][16][17], transportation hierarchy [18], distance table, and separator-based methods [19][20][21] are generally used [6][22][23][24][25][26].

For multi-criteria path planning, some researches focused on finding the pareto-optimal set of paths ideal for all requiring features [27][28]; however, it is time-consuming and unclear to obtain the subset of optimal paths efficiently for multiple origin states in the planning space [29]. Moreover, some of the relevant urban features (i.e. the entropy of urban functions, or the relationship between new and existing routes) generalized in inferring the PF are not superimposable, which makes the heuristic algorithms inappropriate for our case. In other words, a heuristic shall be monotonic so that its estimate is always less than or equal to the estimated cost from any neighboring vertex to the goal plus the cost of reaching that neighbor. However, our input features and its heuristic are not monotonic. Therefore, using these speedup techniques mentioned above, along with the concept of Gaussian mixture model, which is mostly utilized in solving background modelling for real-time tracking problem [30], we propose a target-prioritized searching algorithm.

III. Preliminaries

Definition 1: *Grid*. We divide the city into disjointed grids (0.1km×0.1km) [31] and store all features that are correlated with PF (e.g. population in this grid, whether existing routes passed this grid, etc.) into corresponding grid.

Definition 2: *Grid-like graph*. Grid-like graph is composed of disjointed grids that records connections as original road network based on OpenStreetMap (OSM). Each grid stores the connections between adjacent grids in its eight directions if there exists a road in OSM that connects each other.

Definition 3: *Station*. Station is a facility or area for passenger to regularly get-into or get-off the mass transit transportation. (Note that the mass transit transportation here refers to city bus, light rails, trolley bus, etc.) Passenger need to pay by smart card when getting-into or/and get-ting-off the mass transit at a station. Station in original mass transit data or as input given by users as system is a point with latitude and longitude; but is labelled on a grid that the point located at in grid-like graph.

Definition 4: *Trajectory*. Trajectory is the path that certain mass transit takes. Trajectory in original mass transit data or as input given by users as system is a series of road junctions; but turns into a series of connecting grids in the graph-like grids for further PF inference and route recommendation.

Definition 5: *Route*. Route is a set of combination of trajectory and stations. Note that same series of trajectories with different set of stations does refer to different route. Route is a series of connecting grids and several grids la-belled as stations in the grid-like graph; however, since we divide the city into disjointed grids with a meticulous size, the actual route in real world (OSM) can be easily reproduced given a sequence of grids. Therefore, though some re-projections from grid to actual road network are needed, no other superfluous process needed to handle in post-processing.

Definition 6: *Passenger Flow (PF)*. Since the price is fixed fare for Chicago and Tainan bus transit system, and most of mass transit transportation system in other cities, the passenger flow along the route here refers to the total passengers who passed any point along the route. To be more specific, PF is counted once someone pay by smart card when getting-into or getting-off the mass transit at a station of a route.

Definition 7: *Route affecting region (RAR)*. The demand for public transportation is not only based on the origin and destination, but also the nearby geographical environment and urban functions of nearby areas [5]. Thus, we exploit RAR for considering PF-related features. Circles are first draw from each grid of given route, and then a RAR is formed by this set of circles. The radius of the circles is 400 meters, which is the

walking tolerance for pedestrians based on Design Manual for Urban Sidewalks [32], we then extract features of corresponding grids within RAR for PF inference. Note that the proposed algorithm is applicable with feature extraction of the PF inference module based on either RAR or Origin-Destination (OD) Matrix [33][34].

IV. PF Inference Module

A. Problem Definition

Given a set of trajectories labelled with stations, the goal is to infer the PF for that route.

Since features are already stored in grids, the feature set for each existing route is extracted and integrated as training data along with its corresponding ticket data, which is associated with the timestamp and PF for each route. We adopt DNN for regression as the inference model and treat various features as inputs while PF values as the predictive label. Finally, the pre-trained model is utilized in route planning process to infer PF value of given route. The procedure for the PF inference module is illustrated in Fig. 2.

B. Feature extraction based on RAR

To infer the PF value of a trajectory correctly, we consider six kinds of relevant urban features in RAR:

POI-related features. Various POIs (Point-Of-Interest, refers to specific point location such as transportation hubs or entertainment venues) and their density in RAR indicate the function of a region. The POI entropy shows the diversity of purpose when people visit the nearby area of a route and is based on Information Theory [35]:

$$\text{Entropy}(l_i) = -\sum_{\gamma \in \Gamma} \left(\frac{N_\gamma(l_i,r)}{N(l_i,r)} \times \log \frac{N_\gamma(l_i,r)}{N(l_i,r)} \right) \quad (1)$$

Where Γ indicates set of POI, and γ refers to certain type of POI. Besides, $N(l_i, r)$ displays the total number of POI in RAR of trajectory l_i based on radius r, $N\gamma(l_i, r)$ displays the number of type-γ POI in RAR of trajectory l_i.

Human mobility. The total number of pick-up and drop-off records of Taxi (in Chicago) or Bike (in Tainan) trip records that occur in RAR are accumulated as the leaving and incoming flow respectively. Records taking place in same RAR are viewed as transition flow. Dividing these values by number of all taxi records, several floating-point numbers are derived.

Road network structure. Based on grid-like graph, the degree and closeness centrality in RAR are calculated as floating-point numbers. Degree centrality identifies the total number of reachable vertices for all intersections in RAR, and closeness centrality shows the average distance between one intersection to another in RAR.

Competition and transference with existing routes. To quantify competitive relationship and since the road network structure is already reconstructed into grid-like graph, grids that holds designated routes and each existing route can be labelled respectively. Then a simple algorithm is run to calculate the number of grids that are labelled as both designated route and certain existing one, or grids that are labelled as designated route but nearby grids in its RAR are labelled as certain existing route. Through this process, the numbers of overlap grids and nearby grids between given route and each existing route are derived; meanwhile, if certain existing route is transferable, the grids (except for overlap and nearby ones) of that existing route are viewed as extended grids. Dividing these values by total grids of corresponding existing route, several floating-point numbers are derived, representing overlap/ nearby/ extended region between given and each existing route respectively

Population structure. People in RAR of different ages and genders have different intentions for taking public transportation. Consequently, based on the population pyramid for each village, Population in certain RAR of different ages and genders are derived as several floating-point numbers.

Time information and granularity. Seasons and holidays can influence the passenger flow of public transportation. We adopt one-hot encoding to record the time information for each ticket record.

C. Inference model construction

We adopt *DNN for Regression* to derive the PF for the given route. The input data includes all the features extracted based on the RAR of the given route, while the output is the inferred PF value of that route. The architecture of DNN for Regression is a feed-forward neural network with many levels of non-linearities. Meanwhile, all our input features are rescaled to 0~1, the type of the hidden units for 4 dense layers is ReLU and the output unit is linear.

As the features been considered include relationship with existing routes and other relevant factors as described above, we believe that the model is trained/adapted to infer the PF of an arbitrary route (either new one or existing one) with the features that are extracted based on RAR of the route.

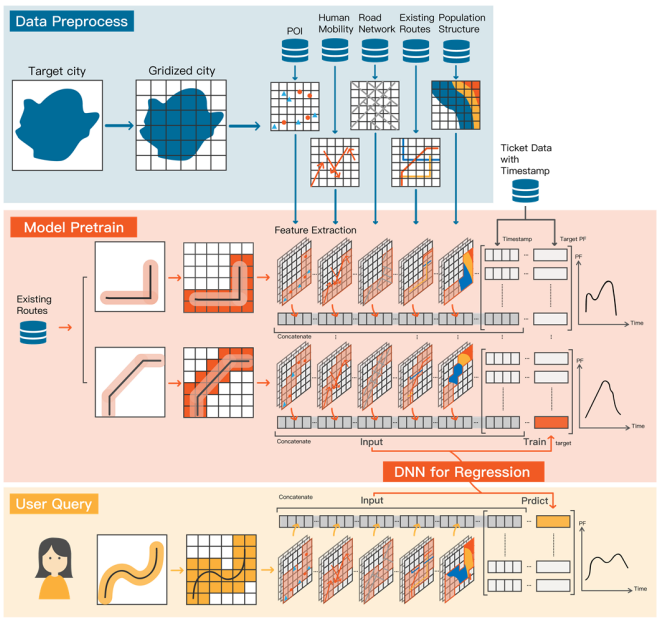

Fig. 2. The procedure for PF inference module. Which is composed on three parts: data preprocessing, model pretrain, and user query. The model chosen to infer PF could be replaced with any regression-based method including but not limited to DNN, SVR, XGBoost, Linear Regression. The user query here also refers to queries generated by different algorithms in the section V.

V. High PF Route Recommendation

A. Problem Definition

Given constraints consist of (1) range for planning (2) a set of must-visit stations $S_M = \{S_{M0}, \ldots, S_{Mi}\}$ (3) the number of recommended stations r (4) maximum length. Our goal is to recommend a trajectory in the given area along with a set of stations $S=S_M+S_R$ to (1) **maximize the inferred PF value** and (2) **minimize the length for the route** (combination of trajectory and stations), where S_M refers to the set of must-visit stations, S_R is the recommended stations $\{S_{R0}, \ldots, S_{Rr}\}$.

B. The Strategy

The optimal solution is quite difficult to obtain in a large urban space due to numerous combinations of trajectories and stations for forming a route. According to our experiments, some exhaustion-based methods are not feasible due to high execution time. Meanwhile, considering the non-monotonic characteristics of input features and heuristic function based on PF, this problem turns out to be NP-Complete and most of heuristic algorithms are inappropriate for our case [36]. Therefore, based on the idea of parallel computing of bidirectional search and goal-directed priority from best-first search, we propose the Multi-source **Bi**directional G**a**ussian-Prioritized **Span**ning Tree (**BiasSpan**) to help retrieve a decent solution in a reasonable time frame. The framework is shown in Fig. 3.

Gird-like graph construction. BiasSpan relies on grid-like graph to search possible routes. Grid-like graph can retain the connectivity of the original road network and merges vertices into fewer grids. To conclude, the benefits is three-fold. First, the number of nodes can be reduced from 390,509 and 237,866 (in the original road network) to 91,320 and 94,282 for two cities respectively since only these grids contain road segments. Second, given a route, the computational loading and time of extracting features in RAR can be saved due to the grid indexing. Third, the grid is clear and easy to visualize for both user queries and route recommendation in map.

PF precomputation and lookup table construction. Since a route consists of multiple connected girds, we first pre-infer the PF value for each grid in the map area and store into a lookup table. Another lookup table is constructed for each grid to record the connectivity between the grids. Since the distance between each grid is either 0.1km or $0.1\sqrt{2}$km, there is no need to store the edge weights (length).

Multi-source bidirectional spanning. Multiple queues are maintained and kept being merged when frontier of one goes to a grid that has been visited by the other queue. We also set a constraint in which one source could only connect to a certain number of sources to prevent from constructing a radial route. The algorithm will repeatedly merge queues until there is a queue that contains all the sources.

Gaussian mixture model for modeling spatial influence. We adopt the idea of Gaussian mixture model (GMM) in two ways. First, for each must-visit station or recommended one, we model its spatial influence (the attractiveness of inferenced passenger in spatial aspect) on other grids using two-dimensional Gaussian distribution. Therefore, for each grid in the graph, we can compute its gained Gaussian distributed PF value from each station. That is, properties of nearby grids are somewhat propagated with a strictly decreasing function based on physical distances to complement the lack of specific information. This distribution smooth a non-monotonic function, which decreases the incorrectness in the number of non-monotonic features when a grid is selected.

Second, GMM is useful for selecting deployed stations from nodes. Assuming we only select locations which have high PF values as recommended stations, it might lead to settle several stations in a very small region since the grids in that region have high inferenced PF values. To avoid this herding effect, we propose to subtract the accumulated Gaussian distributed PF value from the pre-inferenced PF value of each grid. The calculated PF value is the expected gained passengers considering negative effects from other locations. Then, stations could be recommended based on these re-calculated PF values through an iterative process until the number of recommend stations S_R is satisfied. Thus, each recommended station selected is not only based on high inferenced PF value of the grid itself, but also are kept away from each other by adopting this strategy.

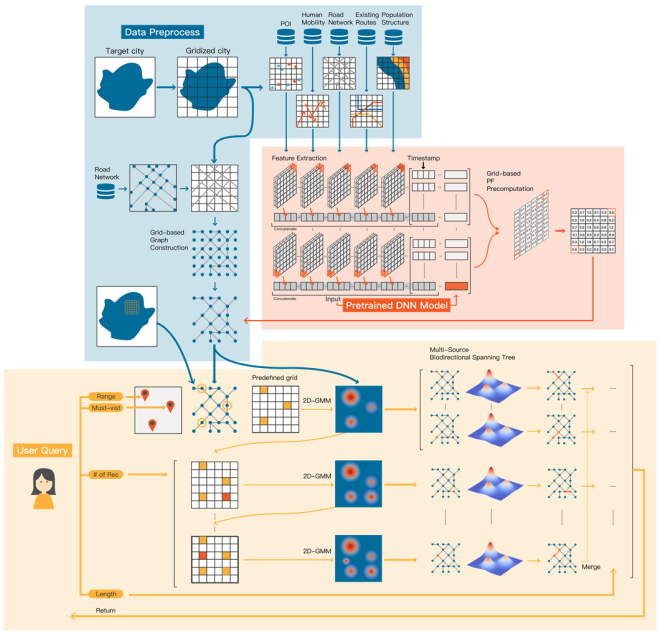

Fig. 3. The procedure for high PF route recommendation. Which consists of three parts with algorithm description depicted in section V-C.

C. Algorithm Description

BiasSpan is composed of three parts: grid-preprocessing, station-recommending, and trajectory-routing.

In grid-preprocessing, it calculates the PF value of each must-visit grid utilizing the PF inference module proposed in previous section; then it evaluates its spreading impact on other grids in the area based on Gaussian function in two dimensions. To be more specifically, several independent Gaussian distributions representing potential PF of each grid are involved in our proposed BiasSpan, where the variance setting of Gaussian distribution could refer to the grid size.

Second, based on the negative Gaussian feedback of inferred PF from must-visit and selected stations, the scores of the GMM

for all grids are derived and we greedily select the grid with maximal PF as the recommended station.

Finally, BiasSpan minimizes depth of search by performing multi-source bidirectional search and prunes the breadth of search space on the basis of the spreading impact of positive Gaussian feedback from other stations, which makes it act as a breadth-first-based, target-prioritized spanning tree growing from multiple stations simultaneously. To ensure a route that connects all stations could be form completely, we allows each starter to grow parallelly until all stations could be connected once and not repeated.

Summary of the properties of BiasSpan.

(1) BiasSpan works for gain function with non-monotonic features based on Gaussian mixture model (GMM).

(2) Vertices in road network structure are contracted into grid-like graph to be fetched directly through the primary key.

(3) Based on GMM with negative feedback, stations are recommended with high PF preventing from crowding-out effect (herding effect).

(4) Searching space for BiasSpan is pruned by bidirectional approach and goal direction technique.

D. Time Complexity

Before exploiting its time complexity, we could transform the original problem into a multiple choice branching problem. Which is to find a subset A'∈A in a directed graph G=(V,A) and a partition of A into disjoint sets A_1, A_2, …, A_m with the sum of weights in subset A' larger than a given positive integer K such that A' contains no cycles and at most one arc from each partition. By reducing into a 3-SAT problem, this "multiple choice branching" problem turns out to be NP-Complete [37] [38]. To be more specific, it remains NP-C even if G is strongly connected and all weights are equal as finding maximum weight branching was viewed as a 2-matroid intersection problem [39]. In other words, a maximum weight branching can be viewed as a maximum weight directed spanning tree for a strongly connected graph.

The time complexity of trajectory-routing for BiasSpan ends up in ***O(EV)*** since the worst case is to traverse each grid in all directions, where *E* refers to the number of maximum directions one spanning tree could try (equaling to branching factor in the worst case/setting), and *V* indicates the number of grids labelled with the road segment in the given area. The schematic search space of Dijkstra's algorithm [40], bidirectional search, and BiasSpan is shown in figure 3, where BiasSpan visits fewer grids than bidirectional as the target-prioritized approach restricts the breadth based on the tendency to other targets.

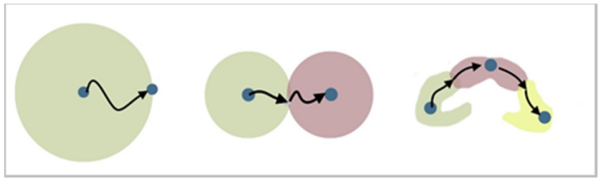

Fig. 4. Schematic search space of Dijkstra's algorithm (left), bidirectional search (middle), and BiasSpan (right).

VI. Evaluation

A. Datasets

We use bus-ticket data on two different types (radial and square structure) of public transit networks from Tainan City Government and Chicago Transit Authority (CTA). The data for Tainan lists ticket id, route id, timestamps, and the starting and ending stations; on the other hand, the data for Chicago lists route id, timestamps, and number of passengers. The datasets contain 14,336,226 and 231,196,847 ticket records respectively and hold at least 100 routes and thousands of stations in service. Public transit networks for both cities are illustrated in Fig. 5.

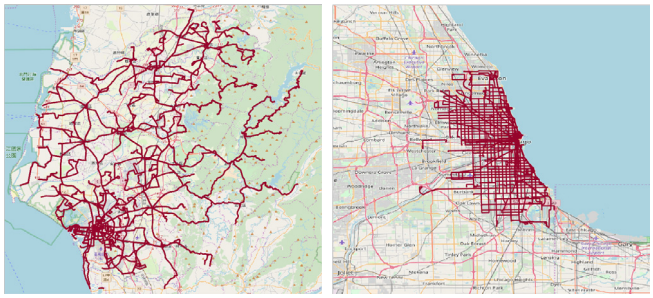

Fig. 5. Public transit networks for Tainan (left) and Chicago (right) on the same scale.

The urban spaces of Tainan and Chicago are divided into 505,296 and 94,282 disjointed grids (0.1km × 0.1km) based on EPSG: 3857, which is a variant of the Mercator projection and acts as the standard for web mapping applications. Since the unit for this projection is meter, we are able to divide the urban spaces into disjointed grids based on meters precisely considering the ellipsoidal datum when generating grid-like graph and pre-processing relevant features in the Geographic Information System. Meanwhile, only 94,282 and 91,320 grids (vertices) would be considered in route recommendation, which are reduced by about three times compared to the original road network structure. On the other hand, static features including POI and road network structures are extracted from GoogleMap and OpenStreetMap. The population are fetched from respective agencies. Finally, we take bike trips and taxi trips that list pick-up and drop-off location as human mobility. Details of both cities are presented in Table I.

TABLE I. Size figures for our input instances

Instance \ Dataset		Tainan	Chicago
Bus data	Existing routes	104	139
	Existing stations	6.575	11,592
	Ticket records	14,336,226	231,196,847
	Period	01/2017 – 12/2017	11/2017 – 10/2018
Gridized map	Grids (0.1km × 0.1km)	505,296	330,335
	Grids labelled with road	94,282	91,320
Other features	POI	8,734	21,889
	Bike trips (for human mobility)	139,478	N/A
	Taxi trips (for human mobility)	N/A	68,461,612
	Road nodes	237,866	390,509
	Road edges	414,409	560,810
	Census blocks (for population)	14,730	46,293

B. Evaluation Setting

We developed six comparative methods to start at one must-visit station and searches for another iteratively: (a) *Dijkstra's Algorithm (Dijkstra's)* [40] (b) *Breadth-First Search (BFS)* [41] (c) *Iterative Deepening Depth-First Search (IDDFS)* [42] (d) *Best-First Search (Best-First)* based on highest pre-calculated PF of the grid [43]. (e) *Distance-Based A* (Distance-A*)* is A* [3] with a heuristic of distance to candidate grid. (f) *Passenger-Flow-Based A* (PF-A*)* is A* [3] with a heuristic that predicts the inferred PF between destination and candidate grid. Baseline method: *Brute-Force (BF)* systematically enumerates all possible combinations and retrieves the optimal one.

Since our optimization-criteria are maximizing PF and minimizing length of route, both results would be reported and an index of "PF per unit length" would be adopted for illustrating the figures. The evaluation is based on 1,000 randomly generated testing cases for given constraints. All methods are implemented in Java and runtime is based on the single core of an Intel i7-7700CPU@3.60GHz with 16 GB of RAM. Considering practical applications, queries executed for over 30 seconds are identified as failures (failure trials would not affect the runtime illustrated in following figures); for methods that have more than half failure cases to respond queries under the parameter setting, their performance would not be considered or displayed in corresponding figures.

TABLE II. STRATEGY SETS TO BE USED IN PRELIMINARY EVALUATION

Strategy \ Set	I	II	III	IV	V	VI	VII	VIII
PF look-up table	✓	✓	✓	✓	✓	✓	✓	✓
Positive GMM	✓	✓		✓	✓	✓	✓	✓
Negative GMM	✓		✓	✓		✓		✓
Multi-source spanning	✓	✓	✓	✓	✓	✓	✓	✓
Bidirectional spanning	✓	✓	✓	✓	✓	✓	✓	
Unidirectional spanning								✓
Best-first spanning	✓	✓	✓					✓
Breadth-first spanning	✓	✓	✓	✓	✓			✓
Depth-first spanning						✓	✓	

C. Evaluation of BiasSpan parameter setting

For this experiment, we run BiasSpan under different standard deviation and negative feedback ratio setting.

The larger standard deviation σ is set, grids in a larger range are distributed but with a lower influence. Results in Fig. 6 show that the ideal setting for σ depends on the size of the mass transit system structure. For instance, since the route network in Tainan is much sparser than the network in Chicago, the distance between station to station is thereafter farther. In other words, the σ in Tainan shall not set to a too small value. On the contrary, the mass transit system in Chicago holds a tiny but closer network structure, experimental results show that a huge drop in performance and runtime takes place when the σ is set to higher than 15.

Besides, experimental results show that negative feedback in the recommending process does affect the quality of the route. More importantly, a huge drop in overall performance takes place when this negative feedback ratio is set to higher than 1.5. To conclude, an ideal setting for negative feedback ratio to effectively prevent herding effect would be 0.6 to 1.

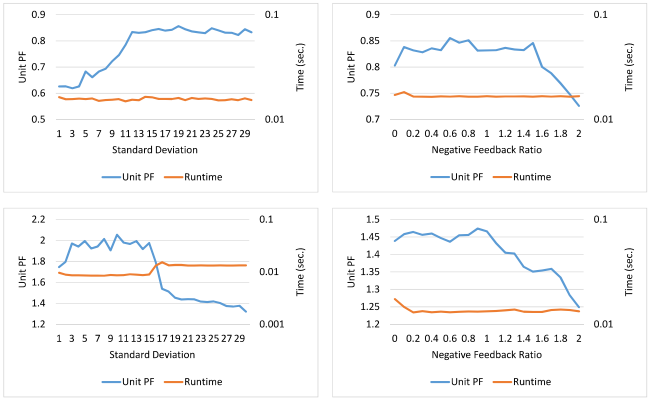

Fig. 6. Unit PF and runtime for different standard deviation and negative feedback ratio settings in Tainan (up) Chicago (down) dataset.

D. Evaluation of strategy selection

To compare the importance of different components of strategies in the proposed algorithm, Table II shows the components of strategy sets based on multi-source spanning for performance comparison. All strategy sets are run under different area ranges and results are summarized in Fig. 7, where the PF per unit length for comparative strategy sets are divided by the value of the strategy set *I* into a unit PF ratio.

First, focusing on the strategy of utilizing GMM with negative feedback in preventing herding effect, strategy set *I* outperforms set *II* and set *III* which only utilized either positive or negative GMM in PF routing process based on the concept of Best-first spanning. Compared to strategy *VII*, strategy *VI*, where Depth-first spanning is adopted, soon fails to construct routes and to return the results for half of the trials. We conclude that negative GMM in PF routing could help to lead the ideal direction into a prior order, but the direction of the first order may not be the best choice when constructing the final route from the entire perspective considering increasing length.

Next, focusing on the choice of spanning strategy: when both positive and negative GMM are considered in PF routing process, strategy set *I* outperforms set *IV* twice where the latter leaves the Best-first spanning and maintains the Breadth-first concept. Furthermore, when Depth-first spanning is adopted, strategy set *VI* fails to construct routes and return the results for half of the trials. Similar results are observed when strategy set *II*, *V*, and *VII* are compared, where the only difference between these methods is the spanning strategy. We conclude that since the maximum number of branches that one grid could try is fixed and lowered to the average branching factor, Best-first spanning strategy does help to construct a better route compared to simply Breadth-first or Depth-first.

On the other hand, utilizing unidirectional spanning strategy instead of bidirectional, strategy set *VIII* (tree search) consumes more time than all other comparative strategy sets (graph search). To be more specific, since the searching space of unidirectional spanning diverges from only single stations, the number of grids that are visited grows rapidly before finding another station.

To conclude, negative feed-back for GMM in PF routing process combined with Best-first and bidirectional spanning outperforms other strategy sets.

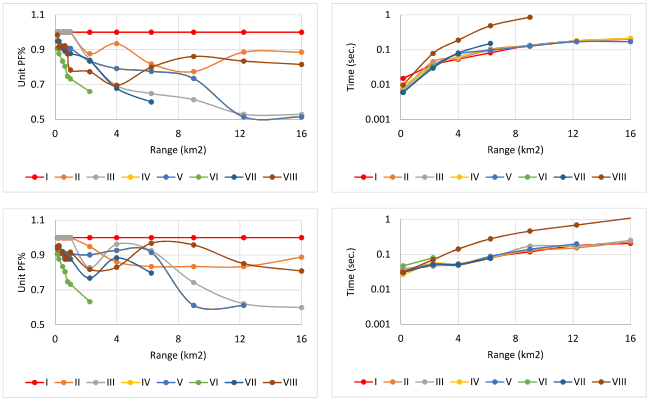

Fig. 7. Unit PF ratio and runtime for different strategy sets under area range in Tainan (up) and Chicago (down) dataset.

E. *Evaluation with Comparative Methods*

For this experiment, we run BiasSpan and all comparative methods for trajectory-routing under different user-constrained settings. To be more specific, this experiment focus on whether a method could form a route that connect same set of stations with a better trajectory. In this section, PF per unit length for each method is divided by the value of BiasSpan into a unit PF ratio. We then compare the PF ratio and runtime of BiasSpan with comparative methods by varying area range and the number of must-visit stations. Results are shown in Fig. 8 and Fig. 9.

First, Fig. 8 shows that proposed BiasSpan performs close to optimal solutions in small space and maintains with high PF per unit length and success rate in the large-scale region. Although *IDDFS* gains better PF results dealing with requests in some cases, the runtime of *IDDFS* is larger than BiasSpan for at least one logarithmic scale. As the main difference between BiasSpan and *IDDFS* is that the latter is a DFS-based algorithm resulting in a higher chance of exploring regions where no station exists. Results in Fig. 9 show that given a large number of requested must-visit stations, IDDFS fails to construct routes for more than half of trials.

Compared to *Distance-A** and *PF-A**, BiasSpan tends to visit more grids (route length) but also results in higher PF; BiasSpan ends up in better overall performance. To sum up, the proposed BiasSpan outperforms other comparative methods from 7% to 22% in a large scale (>9 km^2) case and reasonably close to optimal solutions where *Brute-Force* is able to obtain.

Experimental results on the runtime also meet our estimate that the time complexity for BiasSpan is mainly related to the area range. Compared to other comparative algorithms, the scale of runtime for BiasSpan is close to *Best-First Search* and *Distance-A**. To conclude, proposed BiasSpan outperforms other comparative methods from 7% to 24% in large scale (>9 km^2) and reasonably close to optimal in small space. Besides, *Brute-Force* can obtain optimal solutions but is only feasible for very small ranges (<1 km^2)

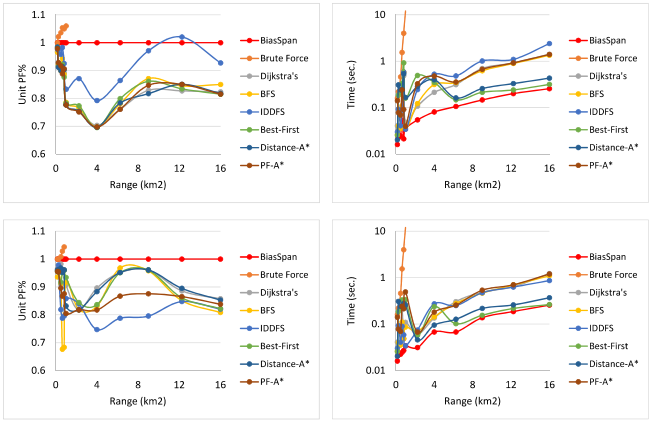

Fig. 8. Unit PF ratio and runtime for different methods under area range in Tainan (up) and Chicago (down) dataset.

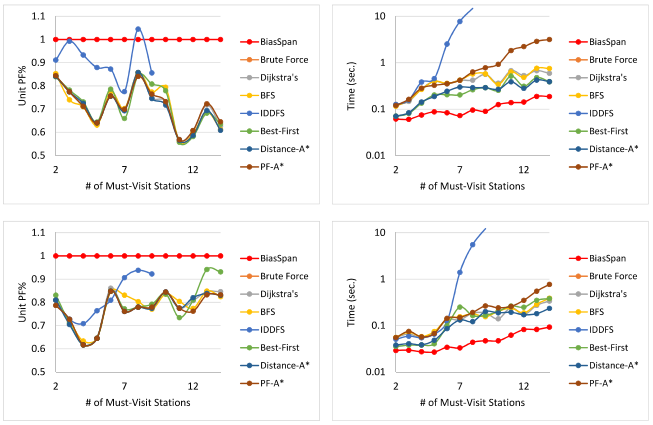

Fig. 9. Unit PF ratio and runtime for different methods under number of must-visit stations in Tainan (up) and Chicago (down) dataset.

VII. Conclusion

Based on data and heuristic derived from neural network, this paper propose a Gaussian-based-prioritized bidirectional searching algorithm - BiasSpan in solving non-monotonic multi-criteria constrained route planning problem. We focus on the application of deploying additional transportation route in existing public transit system given constraints including must-visit stations and area range. Experimental results on mass transportation system in Tainan and Chicago cities show that the spatial influence modelled by GMM does address the non-monotonicity produced by neural networks, and the proposed negative feedback along with GMM in recommending process effectively prevents herding effect. Meanwhile, the proposed BiasSpan outperforms comparative methods from 7% to 24% and runs in reasonable time compared to state-of-art route-planning algorithms.

Acknowledgement

This work was partially supported by Ministry of Science and Technology (MOST) of Taiwan under grants 108-2221-E-006-142 and 108-2636-E-006-013. Meanwhile, we are grateful to Tainan City Government for providing the bus ticket data. Finally, we thank the anonymous reviewers for their careful reading of our manuscript and their many insightful suggestions.

REFERENCES

[1] N. Keng, and Y.Y.D. Yun, "A planning/scheduling methodology for the constrained resource problem". In *Proceedings of the 11th international joint conference on Artificial intelligence* (2): 998-1003. 1989.

[2] K. Osanlou, C. Guettier, A. Bursuc, T. Cazenave, and E. Jacopin, "Learning-based Preference Prediction for Constrained Multi-Criteria Path-Planning". *ICAPS 2019 Scheduling and Planning Applications Workshop (SPARK)*. 2019.

[3] P.E. Hart, N.J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths". *IEEE Transactions on Systems Science and Cybernetics* (4)2: 100–107. 1968.

[4] D.W. Etherington, S. Kraus, and D. Perlis, "Nonmonotonicity and the scope of reasoning". *Artificial Intelligence* 52(3): 221-261. 1991.

[5] F. Lin, and H.-P. Hsieh, "An intelligent and interactive route planning maker for deploying new transportation services". In *Proceedings of the 26th international conference on advances in geographic information systems(SIGSPATIAL)* 620-621. 2018.

[6] D. Delling, P. Sanders, D. Schultes, and D. Wagner, "Engineering route planning algorithms". *Algorithmics of Large and Complex Networks*. Lecture Notes in Computer Science, Vol. 5515. Springer, 117–139. 2009.

[7] I. Abraham, D. Delling, A. Goldberg, and R. Werneck, "A Hub-Based Labeling Algorithm for Shortest Paths in Road Networks". In *Proceedings of the 10th International Symposium on Experimental Algorithms*. Vol. 6630. Springer, 230–241. 2011.

[8] A.V. Goldberg, "A Practical Shortest Path Algorithm with Linear Expected Time". *SIAM Journal on Computing* 37(5): 1637–1655. 2008.

[9] D. Julian, P. Thomas, and W. Dorothea, "User-Constrained Multi-Modal Route Planning". *Journal of Experimental Algorithmics* 19(3):1.1-1.19. 2015.

[10] R.C. Holte, A. Felner, G. Sharon, and N.R. Sturtevant, "Bidirectional Search That Is Guaranteed to Meet in the Middle". In *Proceedings of the 30th National Conference on Artificial Intelligence* 3411-3417. 2016.

[11] J. Chen, R. Holte, S. Zilles, and N.R. Sturtevant, "Front-to-End Bidirectional Heuristic Search with Near-Optimal Node Expansions". In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)* 489-495. 2017.

[12] E. Shaham, A. Felner, N.R. Sturtevant, and J.S. Rosenschein, "Minimizing Node Expansions in Bidirectional Search with Consistent Heuristics". In *Proceedings of the 11th International Symposium on Combinatorial Search (SoCS)* 81-98. 2018.

[13] N.R. Sturtevant, and A. Felner, "A Brief History and Recent Achievements in Bidirectional Search". In *Proceedings of the 32nd National Conference on Artificial Intelligence (AAAI)* 8000-8007. 2018.

[14] A.V. Goldberg, and C. Harrelson, "Computing the shortest path: A* search meets graph theory". In *Proceedings of 16th Annual ACM–SIAM Symposium on Discrete Algorithms*. 156–165. 2005.

[15] T. Yoshizumi, T. Miura, and T. Ishida, "A* with Partial Expansion for Large Branching Factor Problems". In *Proceedings of the 17th National Conference on Artificial Intelligence* 923-929. 2000.

[16] D. Wagner, T. Willhalm, and C. Zaroliagis, "Geometric containers for efficient shortest-path computation". *Journal of Experimental Algorithmics* 10, 1.3: 1–30. 2005.

[17] B. W. Thomas, T. Calogiuri, and M. Hewitt, "An exact bidirectional A⋆ approach for solving resource-constrained shortest path problems". *Wiley Online Library*. doi: 10.1002/net.21856. 2018.

[18] V. Buchhold, P. Sanders, and D. Wagner, "Real-time Traffic Assignment Using Engineered Customizable Contraction Hierarchies". *Journal of Experimental Algorithmics* 2.4. doi: 10.1145/3362693. 2019.

[19] D. Delling, A.V. Goldberg, T. Pajor, and R.F. Werneck, "Customizable route planning". In P*roceedings of the 10th International Symposium on Experimental Algorithms*, Vol. 6630. Springer, 376–387. 2011.

[20] M. Zhang, L. Li, W. Hua, and X. Zhou, "Batch Processing of Shortest Path Queries in Road Networks". In *Proceedings of Australasian Database Conference: Databases Theory and Applications* 3-16. 2019.

[21] Y.H. Li, H.J. Mao, and Y.M. Qin, "Vehicle Routing Problem with Multiple Time Windows and Batch Splitting Based on Inferior First Bidirectional Search Algorithm". In *Proceedings of the 3 rd International Conference on Electrical, Mechanical and Computer Engineering*. doi:10.1088/1742-6596/1314/1/012116. 2019.

[22] H. Bast, "Car or public transport – two worlds". Efficient Algorithms. Lecture Notes in Computer Science, Vol. 5760. Springer, 355–367. 2009.

[23] H. Bast, E. Carlsson, A. Eigenwillig, R. Geisberger, C. Harrelson, V. Raychev, and F. Viger, "Fast Routing in Very Large Public Transportation Networks Using Transfer Patterns". In *Proceedings of the 18th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, Vol. 6346. Springer, 290–301. 2010.

[24] M. Ehrgott, and K. Klamroth, "Connectedness of efficient solutions in multiple criteria combinatorial optimization". *European Journal of Operational Research* 97(1): 159-166. 1997.

[25] R. Zhang, S.N. Kabadi, and A.P. Punnen, "The minimum spanning tree problem with conflict constraints and its variations". *Discrete Optimization* 8: 191-205. 2011.

[26] Q. Song, M. Li, and X. Li, "Accurate and fast path computation on large urban road networks: A general approach". *PLoS ONE* 13(2): e0192274. https://doi.org/10.1371/journal.pone.0192274. 2018.

[27] D. Delling, J. Dibbelt, and T. Pajor, "Fast and Exact Public Transit Routing with Restricted Pareto Sets". In *Proceedings of the Twenty-First Workshop on Algorithm Engineering and Experiments* 54-65. 2019.

[28] D. Delling, T. Pajor, and R.F. Werneck, "Round-Based Public Transit Routing". *Transportation Science*, 49(3): 591–604. 2015.

[29] B.D. Ziebart, A.D. Dey, and J.A. Bagnell, "Fast Planning for Dynamic Preferences". In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2008.

[30] C. Stauffer, and W.E.L. Grimson, "Adaptive background mixture models for real-time tracking". *IEEE CVPR*. Fort Collins, CO, USA, pp. 246-252 Vol. 2. DOI: 10.1109/CVPR.1999.784637. 1999.

[31] L.A. Silman, Z. Barzily, and U. Passy, "Planning the route system for urban buses". *Computers & Operations Research* 1(2): 201-211. 1974.

[32] H.-M. Su, and C.-C. Kuan, "Planning and design guidelines". In Design Manual for Urban Sidewalks, ch. 4, sec. 1, pp. 1-4. 2003.

[33] A. Peterson, "The Origin–Destination Matrix Estimation Problem — Analysis and Computations". Doctoral dissertation, Department of Science and Technology. Linköping University, Sweden. 2007.

[34] H. Yang, and J. Zhou, "Optimal traffic counting locations for origin–destination matrix estimation". *Transportation Research Part B: Methodological* 32(2): 109–126. 1998.

[35] T.M. Cover, and J.A. Thomas, "Entropy, relative entropy and mutual information". Elements of Information Theory, ch. 2, sec. 1, pp. 12-13. 1991.

[36] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, and R. Werneck, "Route Planning in Transportation Networks". Algorithm Engineering. Lecture Notes in Computer Science, vol 9220. Springer. 2016.

[37] M.R. Garey, D.S. Johnson, and L. Stockmeyer, "Some simplified NP-complete graph problems". *Theoretical Computer Science* 1(3): 237-267. 1976.

[38] M.R. Garey, and D.S. Johnson, "Computers and intractability: a guide to the theory of NP-completeness". W. H. Freeman and Company. Appendix B. 1979.

[39] M.R. Garey, D.S. Johnson, and R.E. Tarjan, "The Planar Hamiltonian Circuit Problem is NP-Complete". *SIAM Journal on Computing* 5(4), 704–714. 1976.

[40] E.W. Dijkstra, "A note on two problems in connexion with graphs". *Numerische Mathematik*, 269–271. 1959.

[41] C.Y. Lee, "An Algorithm for Path Connections and Its Applications". IEEE IRE Transactions on Electronic Computers EC10(3): 346–365. 1961.

[42] R.E. Korf, "Depth-first iterative-deepening: an optimal admissible tree search". *Artificial Intelligence* (27)1: 97–109. 1985.

[43] J. Pearl, "Heuristics: intelligent search strategies for computer problem solving". Addison-Wesley Longman Publishing Co. p. 48. 1984.