



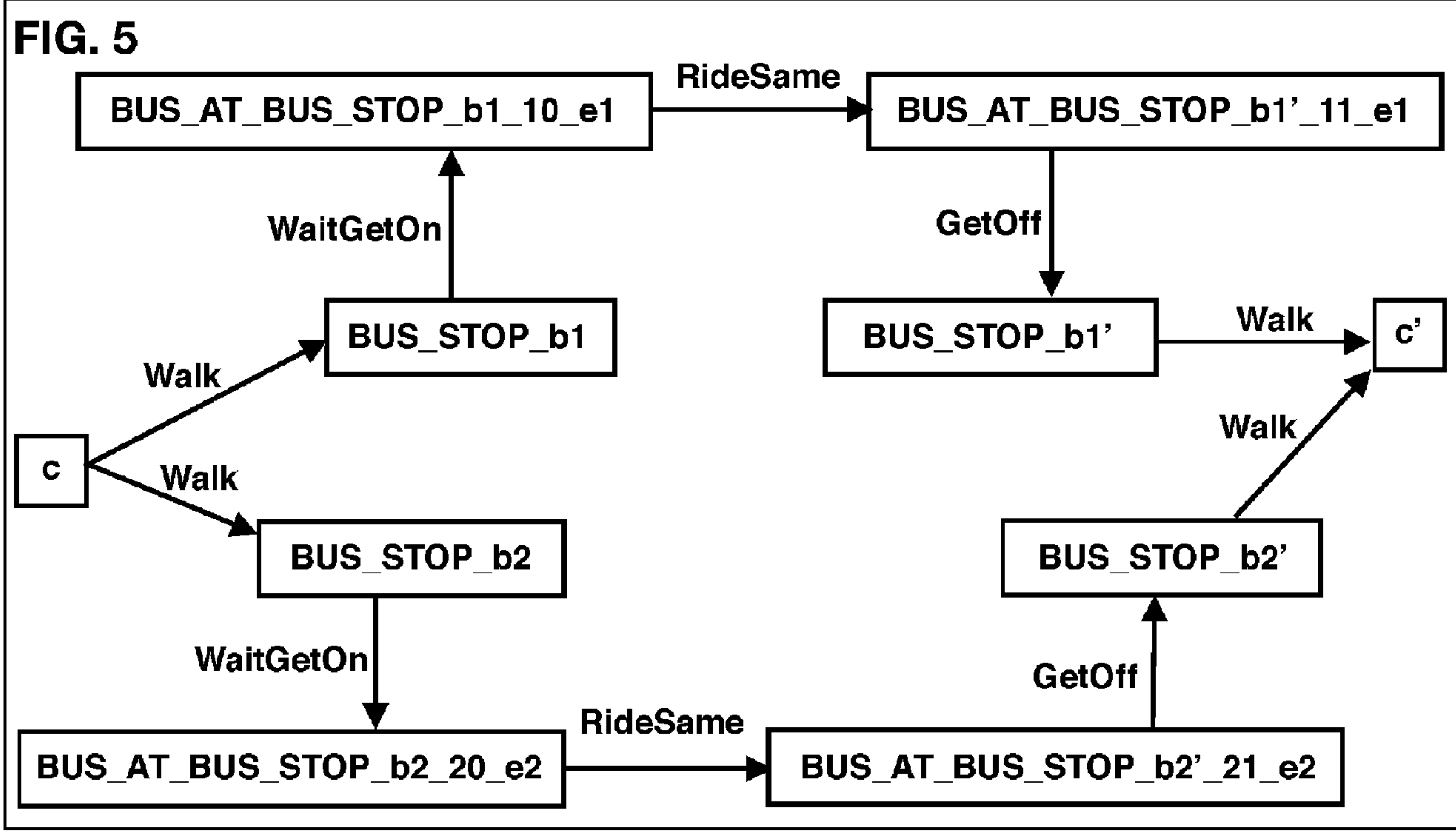
(12) **DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION**

(13) **A1**

(86) Date de dépôt PCT/PCT Filing Date: 2018/11/07
(87) Date publication PCT/PCT Publication Date: 2019/06/27
(85) Entrée phase nationale/National Entry: 2019/10/16
(86) N° demande PCT/PCT Application No.: US 2018/059517
(87) N° publication PCT/PCT Publication No.: 2019/125625
(30) Priorités/Priorities: 2017/12/21 (US62/608,586);
2018/01/05 (US62/613,779); 2018/04/18 (US62/659,157);
2018/04/19 (KR10-2018-0045558);
2018/11/05 (US16/180,050)

(51) Cl.Int./Int.Cl. *G01C 21/34* (2006.01)
(71) Demandeur/Applicant:
MALEWICZ, GRZEGORZ, PL
(72) Inventeur/Inventor:
MALEWICZ, GRZEGORZ, PL
(74) Agent: NA

(54) Titre : PROCEDE ET SYSTEME INFORMATIQUE POUR FOURNIR UN ITINERAIRE OU UNE DUREE
D'ITINERAIRE POUR UN VOYAGE D'UN EMPLACEMENT SOURCE A UN EMPLACEMENT CIBLE
(54) Title: A METHOD AND A COMPUTER SYSTEM FOR PROVIDING A ROUTE OR A ROUTE DURATION FOR A
JOURNEY FROM A SOURCE LOCATION TO A TARGET LOCATION



(57) Abrégé/Abstract:
Embodiments relate to producing a plan of a route in a transportation system. The method receives route requirements, including a starting and an ending locations. The method builds a model of the transportation system from data about vehicles. The model

(57) **Abrégé(suite)/Abstract(continued):**

abstracts a "prospect travel" between two locations using any of a range of choices of vehicles and walks that can transport between the two locations. Given anticipated wait durations for the vehicles and their ride durations, the method determines an expected minimum travel duration using any of these choices. The method combines the expectations for various locations in a scalable manner. As a result, a route plan that achieves a shortest expected travel duration, and meets other requirements, is computed for one of the largest metropolitan areas in existence today. Other embodiments include a computer system and a product service that implement the method.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property
Organization
International Bureau(43) International Publication Date
27 June 2019 (27.06.2019)(10) International Publication Number
WO 2019/125625 A1(51) International Patent Classification:
G01C 21/34 (2006.01)(21) International Application Number:
PCT/US2018/059517(22) International Filing Date:
07 November 2018 (07.11.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

62/608,586	21 December 2017 (21.12.2017)	US
62/613,779	05 January 2018 (05.01.2018)	US
62/659,157	18 April 2018 (18.04.2018)	US
10-2018-0045558	19 April 2018 (19.04.2018)	KR
16/180,050	05 November 2018 (05.11.2018)	US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:

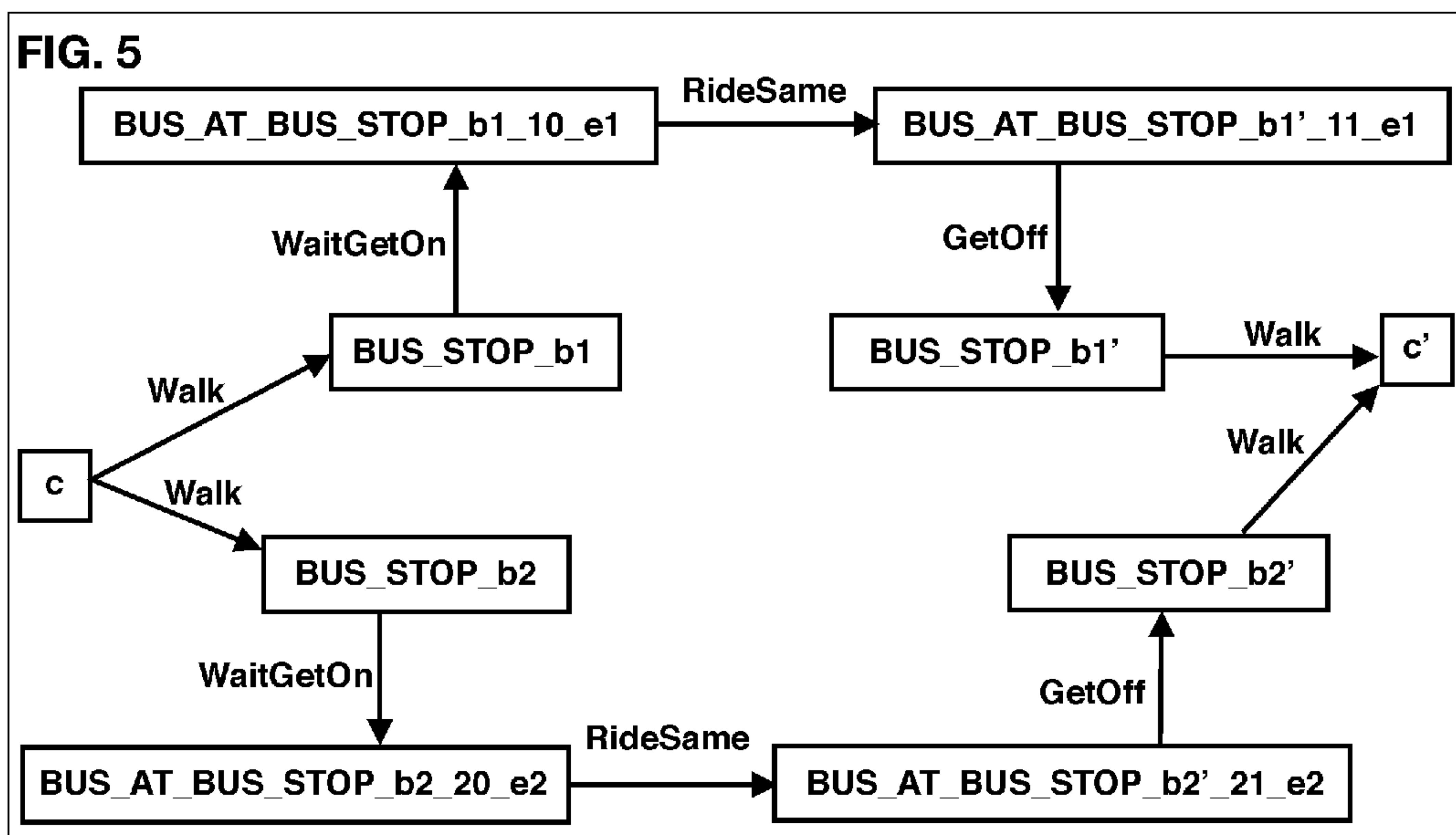
US 16/180,050 (CON)
Filed on 05 November 2018 (05.11.2018)

(72) Inventor; and

(71) Applicant: **MALEWICZ, Grzegorz** [US/—]; Alabastrowa 56, 25753 Kielce (PL).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC,

(54) Title: A METHOD AND A COMPUTER SYSTEM FOR PROVIDING A ROUTE OR A ROUTE DURATION FOR A JOURNEY FROM A SOURCE LOCATION TO A TARGET LOCATION



(57) **Abstract:** Embodiments relate to producing a plan of a route in a transportation system. The method receives route requirements, including a starting and an ending locations. The method builds a model of the transportation system from data about vehicles. The model abstracts a "prospective travel" between two locations using any of a range of choices of vehicles and walks that can transport between the two locations. Given anticipated wait durations for the vehicles and their ride durations, the method determines an expected minimum travel duration using any of these choices. The method combines the expectations for various locations in a scalable manner. As a result, a route plan that achieves a shortest expected travel duration, and meets other requirements, is computed for one of the largest metropolitan areas in existence today. Other embodiments include a computer system and a product service that implement the method.

[Continued on next page]

WO 2019/125625 A1



SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR,
TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

- *with international search report (Art. 21(3))*
- *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

TITLE OF INVENTION

[Title of the invention]

A Method and a Computer System
for Providing a Route or a Route Duration
for a Journey from a Source Location to a Target Location

CROSS-REFERENCE TO RELATED APPLICATIONS

[001] This application is based upon, and claims the priority dates of, applications:

[Country]	[Application Number]	[Filing Date]
USA	62/608,586	December 21, 2017
USA	62/613,779	January 5, 2018
USA	62/659,157	April 18, 2018
South Korea	10-2018-0045558	April 19, 2018
USA	16180050	November 5, 2018

which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

[002] The present invention relates to route planning in a metropolitan area. A goal of route planning is to determine how to travel from one location to other location using the

vehicles available from various providers of transport services. Often it is required for the travel to last as little time as possible, or depart at a certain time, among other requirements. A route typically specifies instructions for a rider, including walk paths and vehicle ride paths.

BRIEF SUMMARY OF THE INVENTION

[003] Embodiments include a method for computing routes, a computer system that implements and executes the method, and a computer service product that allows users to issue routing queries and receive routes as answers.

[004] According to an embodiment of the present invention, a method for generating a route plan is provided. The method receives a query in a form of a source and a target locations of a route, and other requirements that may include a departure time or an arrival deadline. The method builds graphs that model statistical properties of the vehicles. One of the aspects is a “prospect edge” that models travel from a location to other location using any of a range of choices of vehicles and walks. In one embodiment, that edge models an expected minimum travel duration between the two locations. Using a graph, or its extension dependent on specifics of the query, the method generates a route plan as an answer to the query.

[005] According to an embodiment of the present invention, a computer system for generating a route plan is provided. The system is a combination of hardware and software. It obtains information about the transportation system and walks among locations from a plurality of data providers. The system builds a plurality of graphs that model the transportation system, and computes shortest paths in graphs in order to generate a route plan.

[006] According to an embodiment of the present invention, a computer service product for generating a route plan is provided. The service allows a user to specify queries through a User Interface on a device, including a smartphone, and displays generated route plans on the device.

[007] The embodiments of the invention presented here are for illustrative purpose; they are not intended to be exhaustive. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the embodiments.

[008] The data retrieval, processing operations, and so on, disclosed in this invention are implemented as a computer system or service, and not as any mental step or an abstract idea that is disembodied.

[009] In the presentation, the terms “the first”, “the second”, “the”, and similar, are not used in any limiting sense, but for the purpose of distinguishing, unless otherwise is clear from the context. An expression in a singular form includes the plural form, unless otherwise is clear from the context.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWING

[010] The drawings included in the present invention exemplify various features and advantages of the embodiments of the invention:

FIG. 1: depicts a graph G_0 according to an embodiment of the invention;

FIG. 2: depicts a process flow for constructing graph G_0 according to an embodiment of the invention;

FIG. 3: depicts a travel from c to c' involving a walk, a wait, a bus ride, and a walk according to an embodiment of the invention;

FIG. 4: depicts a travel from c to c' involving one bus line but two ways of travel according to an embodiment of the invention;

FIG. 5: depicts two bus lines, each offers a distinct way of travel from c to c' according to an embodiment of the invention;

FIG. 6: depicts a travel from c to c' involving a walk, a wait, a subway ride, and a walk according to an embodiment of the invention;

FIG. 7: depicts a prospect edge for three choices for travel from c to c' according to an embodiment of the invention;

FIG. 8: depicts a prospect edge for two choices for travel from c to c' given duration random variables conditioned on the time of arrival of the rider at c according to an embodiment of the invention;

FIG. 9: depicts pseudocode for computing prospect edges under the interval model of wait durations and fixed travel durations according to an embodiment of the invention;

FIG. 10: depicts how graph $G1$ extends graph $G0$ with edges from sources, when sources in queries are known according to an embodiment of the invention;

FIG. 11: depicts how graph $G2$ extends graph $G0$ with prospect edges to targets, when targets in queries are known according to an embodiment of the invention;

FIG. 12: depicts an example of computing choices for prospect travel continuing from a penultimate stop/station, when the target is revealed at query time according to an embodiment of the invention;

FIG. 13: depicts a process flow of a computer system for answering routing queries according to an embodiment of the invention;

FIG. 14: depicts an example of a rendering of a route in response to a query by a service product on a smartphone of a user according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

4 Detailed description

[011] A metropolitan transport system is composed of vehicles, for example subways and buses. A common goal for a rider is to determine a fastest route from a given location to other location within the metropolitan area. A route can be computed using timetables of the vehicles, and a desired departure time or arrival deadline, as can be seen at major providers of mapping services online. However, in practice some vehicles do not follow timetables exactly, for example due to traffic. In contrast to prior art, the present invention teaches how to compute routes using a mix of vehicles that follow, and not follow, the timetables. The invention utilizes route similarities and wait durations to improve routing results.

[012] Let us illustrate the improvements with a simple example. Consider two consecutive bus stops b_1 and b_2 along the route of a bus. A ride takes 20 minutes on average. Suppose further that the bus arrives at b_1 every 24 minutes on average. For a rider arriving at b_1 at a random time, the average travel duration to b_2 is 32 minutes (wait + ride). Now, suppose that there is other bus that also rides between these two bus stops. Under the same timing assumptions, and assuming independence of the buses, the average travel duration is four minutes shorter. In general, with n buses, the average is $20 + 24/(n + 1)$ minutes. This simply is because the rider can board a bus that arrives at b_1 first.

[013] However, a natural metropolis is more complicated than our simple illustration: there can be sophisticated overlap patterns of routes, with vehicles having differing arrival and speed patterns. It is not even strictly necessary for routes to overlap to achieve improvements, as riders may walk among vehicle stops. The transport system may evolve over time, as subway timetables change and bus routes get added, for example. Besides, we desire an efficient method for computing routes, so as to enable a computer to quickly answer many routing queries even for the largest metropolitan areas.

4.1 Model outline

[014] We introduce a model of a transport system for computing routes or route durations.

[015] We assume that the transport system is composed of two types of vehicles: (1) vehicles that follow fixed timetables, departing and arriving at predetermined times of the day, for example according to a schedule for weekdays; we call this vehicle a *subway*, and call its stops *subway stations*, (2) vehicles whose departure and arrival times are not fixed; we call this vehicle a *bus*, and its stop *bus stops*. Both subway stations and bus stops have fixed geographical locations, so we can determine walks among them. Buses are grouped into *bus lines*. Any bus of a bus line rides along a fixed sequence of bus stops, commonly until the terminal bus stop of the bus line.

[016] In practice, some buses may quite punctually arrive, which may appear to not conform with our model. For example, consider a bus dispatched from the first bus stop of the bus line according to a fixed timetable. The bus may punctually arrive at the first few stops, until reaching an area of the metropolis with unpredictable traffic. When a rider arrives at one of these first bus stops after a subway ride, the wait duration for the bus is

predictable, and so is the total subway-walk-bus travel duration. We can model this case by conceptually adding a *subway* to represent this multi-vehicle subway-walk-bus travel. Similarly, we can conceptualize bus-walk-subway, bus-walk-bus, and subway-walk-subway, and other combinations, when the rides are synchronized or quite punctual. To simplify the presentation of the disclosure, we maintain our assumption of fixed-timetable subways and non-fixed-timetable buses in the rest of this invention description.

[017] Our method is not restricted to routing people by buses or subway. In contrast, our method is more general. It captures many kinds of vehicles that occur in practice. For example, these include: a subway, a bus, a tram, a train, a taxi, a shared van, a car, a self-driving car, a ferry boat, an airplane, a delivery motorbike, a cargo lorry, or a container truck. A route produced by our method can be used to route any object. For example, these include: a person, a cargo, a package, a letter or a food item. We sometimes refer to this object as a rider.

[018] The transport system is modeled through a collection of directed graphs, each consisting of vertices and edges. Each vertex represents a bus stop, a subway station, or an auxiliary entity. Other examples of vertex representations include a train station, a taxi stand, a shared van pickup or drop-off location, a car park, a self-driving car pickup or drop-off location, a platform, a floor, a harbor, a ferry or air terminal, an airport, or a loading dock. Any edge represents a wait, a travel from one to other location, or an auxiliary entity. In one embodiment, an edge has a weight denoting a duration of wait or travel. In other embodiment, a weight is a random variable. In other embodiment, some random variables are conditioned, for example on the time of the day, holiday/non-holiday day type, among others. In other embodiment, some random variables may be correlated with other random variables.

[019] In one embodiment, we determine a probability distribution for a random variable from historical data. For example, we measure how long a bus of a given bus line took to travel from a given bus stop b_1 to a given bus stop b_2 throughout a period of a month, and determine an empirical distribution of travel duration from this one month of samples. In other example, we measure arrival or departure times of a bus of a given bus line from a given bus stop over a period of time, and determine an empirical distribution of a wait time for a bus of the bus line, for each minute of a weekday. In other example, we use a passing

interval reported by a bus operator to determine an average wait time. In other example, a current location of a bus is used to compute a more accurate distribution of a wait duration for the bus.

[020] Some of the random variables used in our method are non-trivial. A trivial random variable has just one value with probability 1. Any other random variable is non-trivial.

[021] A goal of any graph is to help answer any query to find a route or a route duration between any two geographical locations. The starting point of a route is called a *source*, and the ending point called a *target*. The locations are determined by the application of the routing system. For example, the locations could be commercial enterprises, bus and subway stations themselves, arbitrary points in a park, or the current location of a person determined by a Global Positioning System. In one embodiment, we search for a route by applying a Dijkstra's shortest paths algorithm or an A* (A star) search algorithm to a graph, or some adaptations of these algorithms as discussed later.

[022] In some embodiments, we add restrictions on routes. For example, these include: a vehicle type, a vehicle stop type, a threshold on the number of vehicle transfers, a threshold on a wait duration, a threshold on a walk duration, a type of object being routed which may fit in only specific vehicles, a threshold on a monetary cost of travel, a departure time from the source, an arrival time at the target, or a desired probability of arriving before a deadline.

[023] In one embodiment, our method computes routes or route durations that have a smallest expected duration. However, our method is more general. It also computes routes or route durations that are approximately fastest, or that may not be fastest, but that limit the risk of arriving after a given deadline.

[024] Our invention builds several graphs to answer routing queries. Some embodiments extend a graph with extra vertices and edges based on the query.

4.2 Graph G0

[025] The graph called G0 represents routing among bus stops and subway stations. A detailed description of its construction follows. An illustration of a graph G0 is in FIG. 1, and an illustration of the process flow of the construction is in FIG. 2.

4.2.1 Fixed timetables

[026] The first group of vertices and edges represents routing by vehicles that follow fixed timetables. Because timetables are fixed, we can use a known algorithm to compute a fastest route between two locations, that can involve a sequence of multiple vehicles with walks in-between. Hence, we use a single edge to abstract this multi-vehicle travel from a source to a target.

[027] We add vertices that model boarding subway without waiting, as if the rider timed their arrival at a station with a departure of the subway. We introduce two vertices for each subway station s :

$$SUBWAY_FROM_s$$

and

$$SUBWAY_STATION_s.$$

For any two distinct stations s and s' , we have an edge

$$SUBWAY_FROM_s \rightarrow SUBWAY_STATION_s'$$

representing a ride duration from s to s' , possibly involving changing subways and walking (for example from station s , first take subway A to station B, then walk to station C, then take subway D to station s'); the edge is labelled RideManyGetOff. In one embodiment, the weight of the edge is a minimum ride duration during weekday morning rush hours. In other embodiment, we use a random variable for each of many time windows. In other embodiment, the random variable is conditioned on an arrival time of the rider at a location of s , or a departure time from a location of s .

[028] We model an event when a rider arrives at a subway station later during travel, and may need to wait for the subway. For any two distinct stations s' and s'' , we add a vertex

$$SUBWAY_FROM_TO_s'_s''$$

that denotes riding from s' to s'' . There is an edge

$$SUBWAY_STATION_s' \rightarrow SUBWAY_FROM_TO_s'_s''$$

labelled WaitGetOn representing a wait duration to get on a subway traveling from s' to s'' . In one embodiment, the weight of the edge is set to an average wait for a subway that transports the rider to s'' the earliest, given the rider arriving at s' at a random time during weekday morning rush hours. In other embodiment, the weight is set to half of an average interarrival time of any subway from s' to s'' . In other embodiment, we use a random variable for each of many time windows. In other embodiment, the random variable is conditioned on an arrival time of the rider at a location of s' , or a distribution of the arrival time.

[029] We add an edge

$$SUBWAY_FROM_TO_s'_s'' \rightarrow SUBWAY_STATION_s''$$

labelled RideManyGetOff representing a ride duration from s' to s'' possibly involving changing subways and walking. In one embodiment, the weight of the edge is set to an average shortest ride duration during weekday morning rush hours. In other embodiment, we use a random variable for each of many time windows. In other embodiment, the random variable is conditioned on an arrival time of the rider at a location of s' , or a departure time from a location of s' .

4.2.2 Non-fixed timetables

[030] The second group of vertices and edges represents routing by vehicles that do not follow fixed timetables.

[031] For every bus line, we add vertices that model its bus stops, and a bus at the bus stops. The former abstracts a rider outside the bus, the latter a rider inside the bus. Let $b_1, \dots, b_n$ be the n consecutive bus stops along a bus line e (including on-demand stops). Then we add vertices

$$BUS_STOP_b_k$$

and

$$BUS_AT_BUS_STOP_b_k_k_e,$$

for each k , $1 \leq k \leq n$. Two bus lines may share a bus stop. There is an edge

$$BUS_AT_BUS_STOP_b_k_k_e \rightarrow BUS_STOP_b_k$$

labelled GetOff denoting disembarking the bus at this bus stop; the edge has zero weight.

There is an edge in the reverse direction

$$BUS_STOP_b_k \rightarrow BUS_AT_BUS_STOP_b_k_k_e$$

labelled WaitGetOn representing a duration of waiting for a bus of bus line e at bus stop b_k before embarking. In one embodiment, the weight of the edge is set to half of an average interarrival time of a bus of bus line e during weekday morning rush hours, which is the same for every bus stop of that bus line. In other embodiment, we use a random variable for each of many time windows and bus stops. In other embodiment, the random variable is conditioned on an arrival time of the rider at a location of b_k , or a distribution of arrival time.

[032] To model travel inside the same bus, we add an edge

$$BUS_AT_BUS_STOP_b_k_k_e \rightarrow BUS_AT_BUS_STOP_b_{k+1}_k + 1_e$$

labelled RideSame representing a duration of a ride from bus stop b_k to the next bus stop b_{k+1} by bus line e . In one embodiment, the weight of the edge is set to an average ride duration between these bus stops during weekday morning rush hours. In other embodiment, we use a random variable for each of many time windows and bus stops. In other embodiment, the random variable is conditioned on an arrival time of a bus at a location of b_k , or a departure time from a location of b_k .

4.2.3 Walks

[033] We use walks to connect bus stop and subway station vertices.

[034] In this and other sections of the invention disclosure we allow various requirements for walks. In one embodiment, we use a walk with a shortest duration at a specific speed of 4km/h. In other embodiment, the weight is a random variable for each of several walk path requirements, including speed 6km/h, avoid stairs, avoid dark streets. In other embodiment, we allow only walks with duration at most a fixed amount of time, for example one hour. In other embodiment, a walk is straight-line that ignores any obstacles. In other embodiment, a walk can include travel by a lift, a moving path, an elevator, or an escalator.

[035] We add edges

$$BUS_STOP_b \rightarrow BUS_STOP_b',$$

$$BUS_STOP_b \rightarrow SUBWAY_STATION_s,$$

$$SUBWAY_STATION_s \rightarrow BUS_STOP_b, \text{ and}$$

$$SUBWAY_STATION_s \rightarrow SUBWAY_STATION_s',$$

for any b, b', s, s' , when allowed by the requirements. Each edge is labelled Walk, and its weight represents a duration of a walk.

4.2.4 Constraints

[036] Next we add auxiliary vertices that enable modeling a constraint on the first wait along a route. In one embodiment the wait is zero, which models a rider walking to the stop/ station just early enough to catch a departing bus/subway, but not earlier. In other embodiment, the wait depends on a start time of travel, which models a rider starting the travel at a specific time; for example leaving home at 8am.

[037] We cluster bus stops and subway stations based on their geographical proximity. In one embodiment, we fix the cluster radius to 2 meters. In other embodiment, we select the number of clusters depending on a resource/quality trade-off required by the user of the routing system. In other embodiment, the cluster radius is 0 meters, in which case the clusters are simple replicas of bus stops and subway stations.

[038] For each cluster c , we add a vertex

$$STOPSTATION_CLUSTER_SOURCE_c$$

and add edges connecting the cluster to its buses and subways:

$$STOPSTATION_CLUSTER_SOURCE_c \rightarrow BUS_AT_BUS_STOP_b_k_k_e \text{ and}$$

$$STOPSTATION_CLUSTER_SOURCE_c \rightarrow SUBWAY_FROM_s,$$

when b_k or s are in cluster c . The edges are labelled FirstWaitGetOn. In one embodiment, the weight of the edge is 0. In other embodiment, the weight of the edge is a random variable denoting a wait duration for a vehicle (bus e , or subway) conditioned on a time of arrival of the rider at the location of the vertex (bus stop b_k , or subway station s). In other

embodiment, the weight is increased by a walk duration between c and b_k or s , for example when cluster radius is large.

[039] Note that any non-trivial path in the graph from

$STOPSTATION_CLUSTER_SOURCE_c$

will traverse that FirstWaitGetOn edge exactly once.

[040] We add other auxiliary vertices. We cluster bus stops and subway stations similar as before, and for each cluster c , add a vertex

$STOPSTATION_CLUSTER_TARGET_c$

and edges

$BUS_STOP_b \rightarrow STOPSTATION_CLUSTER_TARGET_c$ and

$SUBWAY_STATION_s \rightarrow STOPSTATION_CLUSTER_TARGET_c$,

for every b and s , when in cluster c . The edges are labelled Zero and have weight 0. In other embodiment, the weight is increased by a walk duration, for example when cluster radius is large.

[041] The introduction of vertices

$STOPSTATION_CLUSTER_TARGET_c$

can help decrease the size of the graph when there are many routing target locations. In other embodiment, we can replace these vertices with direct edges from

BUS_STOP_b

and

$SUBWAY_STATION_s$

to the target, for any b and s when appropriate.

[042] The graph constructed so far models a duration of travel from

$STOPSTATION_CLUSTER_SOURCE_c$

308 to

309 $STOPSTATION_CLUSTER_TARGET_c'$,

310 for any c and c' , such that the first bus or subway is boarded without waiting or with given
 311 waiting, and after the rider gets off the bus or the subway sequence, any subsequent vehicle
 312 ride requires waiting to board.

313 4.2.5 Prospect edges

314 [043] Next we add auxiliary vertices and edges that reflect improvements in travel du-
 315 ration due to using *any* of several vehicles. The improvements may be caused by a shorter
 316 wait for any vehicle, or a shorter ride by any vehicle.

317 [044] The duration of waiting to board a vehicle can be modeled by assuming that the
 318 rider arrives at a stop/station at a random time, because of a stochastic nature of the vehicles
 319 that use non-fixed timetables. If there were two consecutive RideManyGetOff vehicle ride
 320 edges on a graph path, the edges could be replaced by one RideManyGetOff edge.

321 [045] We introduce a *prospect edge*, which abstracts travel between two locations using
 322 one of several choices of vehicles. In one embodiment, the weight of the edge is the value of
 323 an expected *minimum* travel duration among these choices.

324 [046] In this section, the two locations connected by a prospect edge are near vehicle
 325 stops. However, this is not a limitation of our method. Indeed, in a later section we describe
 326 a prospect edge that ends at an arbitrary location that may be far from any vehicle stop.
 327 In general, a prospect edge may connect arbitrary two vertices in a graph. However, for the
 328 sake of presentation, in this section we focus on prospect edges near vehicle stops.

329 [047] We cluster bus stops and subway stations based on their geographical proximity,
 330 similar as before. Given two distinct clusters c and c' , we consider any way of traveling from
 331 c to c' by a walk, followed by a bus ride, followed by a walk, any of the two walks can have
 332 length 0. For example, FIG. 3 depicts a case when there is a walk from c to vertex

333 BUS_STOP_b ,

334 and from there a graph path involving bus line e with edges WaitGetOn, RideSame, and
 335 GetOff, ending at a vertex

336 BUS_STOP_b' ,

337 and then a walk from

338 BUS_STOP_b'

339 to c' . Let T be a random variable representing a duration of travel from c to c' using the
 340 walks from c to b and from b' to c' , and a bus ride from b to b' modeled by a graph path.
 341 This variable is just a sum of the random variables of the graph edges along the path, plus
 342 the random variables of two front and back walks. Its distribution can be established from
 343 the constituent distributions. In one embodiment, we condition the random variable on a
 344 departure time from c .

[048] In one embodiment, this random variable T is uniformly distributed on an interval
 $[x, y]$, where the interval tips are

$$\begin{aligned} x &= (\text{minimum walk duration from } c \text{ to } b) \\ &+ (\text{sum of an expected RideSame duration along the path edges}) \\ &+ (\text{minimum walk duration from } b' \text{ to } c'), \text{ and} \\ y &= x + 2 \cdot (\text{expected WaitGetOn duration}). \end{aligned}$$

345 In other embodiment, the tips are adjusted by a multiplicity of a standard deviation of the
 346 random variables. In other embodiment, we consider c, b, b', c' only when the walk durations
 347 from c to b and from b' to c' are at most a fixed amount time, for example one hour. In other
 348 embodiment, any of the walks may be zero-length (an optional walk). In other embodiment,
 349 we require a shortest duration walk from c to b , or from b' to c' . In other embodiment, walks
 350 may have embodiments as in Section 4.2.3. In other embodiment, the random variable T is
 351 non-uniform. In other embodiment, the random variable T is conditioned on an arrival time
 352 of the rider at a location of c .

353 [049] For a fixed bus line e , there may be many alternatives for traveling from c to c' ,
 354 because the rider can board/get off at various bus stops of that bus line, and use walks for
 355 the rest of the travel. For example, FIG. 4 extends FIG. 3 by showing an alternative ride:
 356 to one further stop

357 BUS_STOP_b''

that increases a total ride duration, but decreases a total walk duration. In one embodiment, from among these alternatives, we take a random variable T that has a lowest expectation. Let us denote this variable $T_{c,c',e}$. This is a fixed random variable for the bus line e , and the start and the end clusters c and c' . The variable denotes a fastest travel duration for getting from c to c' by the bus line e stochastically. In one embodiment, when the candidates for $T_{c,c',e}$ are uniformly distributed on intervals, a lowest expectation candidate is just a candidate with a smallest median value of its interval. In other embodiment, we use one variable for each of many time windows, for example so as to capture higher frequency of buses during peak hours, and also higher road traffic. In other embodiment, the random variable is conditioned on an arrival time of the rider at a location of c .

[050] Let us consider all bus lines e_1 through e_n that can help transport a rider from c to c' . Note that the constituent walks and bus stops may differ. For example, FIG. 5 shows two bus lines e_1 and e_2 , each using distinct bus stops, and having different walk durations. Let T_{c,c',e_1} through T_{c,c',e_n} be respective fastest travel duration random variables, as defined before.

[051] We can compute an expected minimum of the variables $E[\min_{1 \leq i \leq n} T_{c,c',e_i}]$. This expectation models travel duration by “whichever bus will get me there faster”. In one embodiment, the random variables of different bus lines are independent. That is T_{c,c',e_i} is independent from T_{c,c',e_j} for any two distinct bus lines e_i and e_j . In other embodiment, the random variables are independent uniform on a common interval $[x, y]$. Then an expected minimum is $(y + n \cdot x)/(n + 1)$. In other embodiment, we compute the expectation through a mathematical formula, approximate integration, random sampling, or other approximation algorithm or a heuristic for an expected minimum. When an approximation algorithm is used, then our method no longer produces shortest routes, but instead produces approximately shortest routes.

[052] Now we discuss how to include subways into a computation of an expected minimum travel duration. Similar to buses, let $T_{c,c',s,s'}$ be a random variable of fastest travel duration from c to c' using walks and subway rides. As illustrated in FIG. 6, there is a walk from c to s , a path in the graph

SUBWAY_STATION_s

388 $\rightarrow SUBWAY_FROM_TO_s_s'$

389 $\rightarrow SUBWAY_STATION_s',$

390 and a walk from s' to c' . A distribution of this variable can be established from constituent
391 distributions. In one embodiment, we condition the random variable on a departure time
392 from c .

[053] In one embodiment, $T_{c,c',s,s'}$ is uniformly distributed on an interval $[x, y]$, where the interval tips are

$$\begin{aligned} x &= (\text{minimum walk duration from } c \text{ to } s) \\ &+ (\text{expected RideManyGetOff duration on the graph path}) \\ &+ (\text{minimum walk duration from } s' \text{ to } c'), \text{ and} \\ y &= x + 2 \cdot (\text{expected WaitGetOn duration on the graph path}). \end{aligned}$$

393 In other embodiment, the tips are adjusted by a multiplicity of a standard deviation of the
394 random variables. In other embodiment we restrict the walk durations from c to s and from
395 s' to c' to at most a fixed amount time, for example one hour. In other embodiment, any
396 of the walks may be zero-length (an optional walk). In other embodiment, we require a
397 shortest duration walk from c to s , or from s' to c' . In other embodiment, walks may have
398 embodiments as in Section 4.2.3. In other embodiment, $T_{c,c',s,s'}$ is non-uniform. In other
399 embodiment, $T_{c,c',s,s'}$ is conditioned on an arrival time of the rider at a location of c . In other
400 embodiment, we use one variable for each of many time windows.

401 [054] A complication arises in that the subway random variables are pairwise dependent,
402 because they are derived from fixed subway schedules. This may complicate a computation
403 of an expected minimum travel duration from c to c' .

404 [055] Let us consider all subway rides that can help transport a rider from c to c' , and
405 let $s_1, s'_1, \dots, s_m, s'_m$ be the m embarkation and disembarkation subway stations with the
406 respective random variables T_{c,c',s_1,s'_1} through T_{c,c',s_m,s'_m} .

[056] In one embodiment, any one subway random variable together with all bus line random variables are independent. In that case we can compute an *expected minimum* travel

duration for buses and subways as a minimum of expected minima, adding one subway ride at a time to the pool of bus rides, and denote it as $P(c, c')$, as in the following equation:

$$P(c, c') = \min_{1 \leq j \leq m} E[\min(T_{c, c', s_j, s'_j}, T_{c, c', e_1}, \dots, T_{c, c', e_n})]. \quad (1)$$

407 We call $P(c, c')$ a *prospect travel*, because it is a travel from c to c' involving any of several
 408 transportation choices, *opportunistically*. We call the $m + n$ constituent random variables
 409 T_{c, c', s_j, s'_j} and T_{c, c', e_i} the *choices*.

410 [057] In one embodiment, T_{c, c', e_i} is uniform over an interval, and so is T_{c, c', s_j, s'_j} . In that
 411 case we compute an expected minimum $E[\min T_i]$, for some number of T_i , each uniform over
 412 an interval $[x_i, y_i]$.

413 [058] For example, FIG. 7 shows travel from c to c' involving three choices:

- 414 • bus line e' with wait uniform on $[0, 900]$ and walk&ride 1700,
- 415 • bus line e'' with wait uniform on $[0, 3600]$ and walk&ride 1000,
- 416 • subway with wait uniform on $[0, 300]$ and walk&ride 2200.

417 In that case a minimum expected travel duration is $2150 = \min\{2150, 2800, 2350\}$, which
 418 does not reflect improvements from travel by “whichever is faster”. However, an expected
 419 minimum is lower: $P(c, c') = 1933$.

420 [059] In other example, FIG. 8 illustrates probability distributions conditioned on a time
 421 when the rider arrives at c (the source of a prospect edge). There are two choices of getting
 422 from c to c' , one by bus and the other by subway. Each choice has its own conditional
 423 probability distributions for wait and for walks and ride.

424 [060] There is a gain in duration due to a prospect travel, if the value of $P(c, c')$ is
 425 less than a minimum of expectations $\min(\min_{1 \leq j \leq m} E[T_{c, c', s_j, s'_j}], \min_{1 \leq i \leq n} E[T_{c, c', e_i}])$. In that
 426 case, we add to the graph: vertices

427 $PROSPECT_CLUSTER_SOURCE_c$

428 and

429 $PROSPECT_CLUSTER_TARGET_c'$,

430 and an edge

431 $PROSPECT_CLUSTER_SOURCE_c \rightarrow PROSPECT_CLUSTER_TARGET_c'$

432 labelled AvgMinWalkWaitRideWalk with the weight $P(c, c')$. We also add edges from bus
433 and subway stations of the cluster c to the vertex

434 $PROSPECT_CLUSTER_SOURCE_c,$

435 and edges from the vertex

436 $PROSPECT_CLUSTER_TARGET_c'$

437 to bus and subway stations in the cluster c' ; these edges are labelled Zero and have zero
438 weight. In one embodiment we add the prospect edge only when its weight $P(c, c')$ results
439 in a gain that is above a threshold, for example at least 10 seconds.

440 [061] We remark that our method does not require the rider to board a first arriving
441 of the transportation choices, simply because a subsequent choice, even though requiring a
442 longer wait, may arrive at the destination faster (consider an express bus versus an ordinary
443 bus). Our method does not even require boarding a bus at the same stop/station, because
444 the rider may walk to other stop/station, for example anticipating an express train departing
445 from there.

446 **Definition 1** [062] *In one embodiment, prospect travel is defined in terms of:*

- 447 • *any two locations c and c' ,*
- 448 • *any number $k \geq 2$ of random variables $T_1, \dots, T_k$, each representing a duration of*
449 *travel from c to c' ,*
- 450 • *the k variables are independent, dependent, or correlated arbitrarily,*
- 451 • *any of the k variables may be conditioned on a time A of arrival of the rider at a*
452 *location c ; the time A may be a random variable.*

453 *The duration of prospect travel is a minimum $\min(T_1, \dots, T_k)$, which by itself is a random*
454 *variable. The weight of a prospect edge is an expected value of this minimum $P(c, c') =$*
455 *$E[\min(T_1, \dots, T_k)]$.*

[063] In other embodiment, a random variable T_i is distributed uniformly on an interval. In one embodiment, a random variable T_i is conditioned on an arrival time at c that falls within a specific time window, or a probability distribution of arrival time at c .

[064] In one embodiment, in order to determine the random variables $T_1, \dots, T_k$, we determine a list of vehicle stops near c and durations of walks to these stops from c , and a list of vehicle stops near c' and duration of walks from these stops to c' , and then for each pair of vehicle stops on the two lists, we determine a travel duration random variable.

[065] In one embodiment, we compute various statistics on a random variable $\min(T_1, \dots, T_k)$. One is the already mentioned expected value. But we also compute a probability mass, which can be used to determine an arrival time that can be achieved with a specific probability. In order to compute these statistics, we use several methods, including sampling, a closed-form formula, approximate integration, and other approximation algorithm or a heuristic.

[066] In one embodiment, we pre-compute a component of prospect travel and store it, so that when prospect travel needs to be determined, we can retrieve the component from storage and avoid computing the component from scratch. Examples of such components include: a random variable of a duration of travel between a pair of vehicle stops; an expected minimum of two or more travel duration random variables; a probability distribution of a minimum of at least two travel duration random variables; or a path or a travel duration between a pair of vehicle stops.

[067] So far we have defined how to compute a prospect edge for a given c and c' . We apply this definition to all pairs of distinct c and c' , which determines which prospect clusters get connected, and which do not get connected, by a prospect edge, and of what weight.

[068] In one embodiment, instead of considering a quadratic number of c and c' pairs, we perform a graph traversal. In one embodiment, we use a “forward” traversal from vertex

PROSPECT_CLUSTER_SOURCE_c,

for each c , towards every vertex

PROSPECT_CLUSTER_TARGET_c'

that is reachable by walk-bus/subway-walk. During this traversal, we identify the graph paths that lead to the

486 $PROSPECT_CLUSTER_TARGET_c'$,

487 for each c' . Once we have identified all such paths for a specific c' , we have computed all the
488 choices between the

489 $PROSPECT_CLUSTER_SOURCE_c$

490 and the

491 $PROSPECT_CLUSTER_TARGET_c'$,

492 and thus can compute an expected minimum of these choices (see FIG. 9 for a further
493 example). Because we limit the exploration to only the reachable parts of the graph, we
494 can often compute prospect edges more efficiently. In one embodiment, we use a symmetric
495 method of a “backward” traversal from vertex

496 $PROSPECT_CLUSTER_TARGET_c'$,

497 for each c' , backwards to every vertex

498 $PROSPECT_CLUSTER_SOURCE_c$

499 that is reachable by a “reversed” path walk-bus/subway-walk.

500 [069] FIG. 9 illustrates an embodiment of the process of adding prospect edges to
501 graph G0, in the case when any wait duration is uniformly distributed on an interval
502 $[0, 2 \cdot \text{WaitGetOn}]$ for the respective edge, and a ride duration is deterministic.

503 4.3 Extensions of graph G0

504 [070] We describe extensions to the graph G0. Each extension is useful for a specific kind
505 of routing queries.

506 4.3.1 Sources known beforehand

507 [071] In some embodiments the source locations of routing queries are known in advance.
508 For example, suppose that we are interested in finding a shortest route from every restaurant
509 in the metropolitan area, and the restaurant locations are known. This can be achieved with
510 the help of an extended graph G0.

511 [072] In one embodiment, for each such source s , we add a vertex

512 $SOURCE_s$.

513 See FIG. 10 for an illustration. In one embodiment, we add an edge from

514 $SOURCE_s$

515 to any bus stop and subway station cluster

516 $STOPSTATION_CLUSTER_SOURCE_c$

517 in the graph G0. The edge is labelled Walk, and its weight represents a duration of a walk.

518 In other embodiment, we use a shortest walk with duration that is at most a threshold, or

519 other embodiments as in Section 4.2.3.

520 [073] The resulting graph is denoted G1 (it includes G0). G1 can be used to compute

521 shortest paths from any

522 $SOURCE_s$

523 to any

524 $STOPSTATION_CLUSTER_TARGET_c$.

525 In one embodiment, some paths are pre-computed, stored, and retrieved from storage when

526 a query is posed.

527 [074] In other embodiment, we use a symmetric method when targets are known before-

528 hand: for each target t , we add a vertex

529 $TARGET_t$,

530 and add an edge from any

531 $STOPSTATION_CLUSTER_TARGET_c$

532 to any

533 $TARGET_t$

534 labelled Walk. The resulting graph is denoted G1'.

4.3.2 Targets known beforehand

[075] In some embodiments the target locations of routing queries are known in advance, and we extend G_0 with prospect edges to the targets.

[076] For each target t , we add a vertex

$$TARGET_t.$$

See FIG. 11 for an illustration. In one embodiment, we add an edge from any bus stop and subway station cluster

$$STOPSTATION_CLUSTER_TARGET_c$$

in graph G_0 to

$$TARGET_t.$$

The edge is labelled Walk, and its weight represents a duration of a walk. In other embodiment, we use a shortest walk with duration that is at most a threshold, or other embodiments as in Section 4.2.3.

[077] We add prospect edges according to a process similar to Section 4.2.5. Specifically, for any

$$PROSPECT_CLUSTER_SOURCE_c$$

and

$$TARGET_t,$$

we determine all paths from c to t of two kinds:

(1) a ride by a bus with walks: walk from c to b , graph path

$$BUS_STOP_b$$

$$\rightarrow BUS_AT_BUS_STOP_b_i_e$$

...

$$\rightarrow BUS_AT_BUS_STOP_b'_j_e$$

$$\rightarrow BUS_STOP_b'$$

$$\rightarrow STOPSTATION_CLUSTER_TARGET_c''$$

$$\rightarrow TARGET_t,$$

562 (2) a ride by subways with walks: walk from c to s' , graph path

563 $SUBWAY_STATION_s'$
 564 $\rightarrow SUBWAY_FROM_TO_s'_s''$
 565 $\rightarrow SUBWAY_STATION_s''$
 566 $\rightarrow STOPSTATION_CLUSTER_TARGET_c'$
 567 $\rightarrow TARGET_t.$

568 In other embodiment, we use a shortest walk with duration that is at most a threshold, or
 569 other embodiments as in Section 4.2.3. We define the random variables of travel duration
 570 along each path just like in Section 4.2.5.

571 [078] In one embodiment, we assume that the kind (1) are independent random variables,
 572 and the kind (2) are dependent. And then we compute an expected minimum travel duration
 573 by considering a pool of all kind (1) random variables (appropriately removing duplicates
 574 for repeated bus lines), adding to the pool one kind (2) random variable at a time, like in
 575 Equation 1 for $P(c, c')$. In other embodiment, we use Definition 1 of prospect travel. This
 576 defines $P(c, t)$, called prospect travel from c to t .

577 [079] When there is gain in travel duration over a minimum of expectations, we add an
 578 edge from

579 $PROSPECT_CLUSTER_SOURCE_c$

580 to

581 $TARGET_t$

582 labelled AvgMinWalkWaitRideWalk with weight $P(c, t)$. We use similar embodiments to
 583 these we used for the edge from

584 $PROSPECT_CLUSTER_SOURCE_c$

585 to

586 $PROSPECT_CLUSTER_TARGET_c'$

587 defined before.

588 [080] In one embodiment, we use a “forward” or a “backward” graph traversal as described
589 in Section 4.2.5 to speed up a computation of prospect edges between

590 $PROSPECT_CLUSTER_SOURCE_c$

591 and

592 $TARGET_t$,

593 for all c and t . In other embodiment, this traversal could be merged into a traversal when
594 computing prospect edges in G_0 .

595 [081] The resulting graph is denoted G_2 (it includes G_0). G_2 can be used to compute
596 shortest paths from any

597 $STOPSTATION_CLUSTER_SOURCE_c$

598 to any

599 $TARGET_t$.

600 In one embodiment, some paths are pre-computed, stored, and retrieved from storage when
601 a query is posed.

602 [082] In other embodiment, we use a symmetric method when sources are known before-
603 hand: for each source s , we add a vertex

604 $SOURCE_s$,

605 and compute a prospect edge from any

606 $SOURCE_s$

607 to any

608 $PROSPECT_CLUSTER_TARGET_c$.

609 The resulting graph is denoted G_2' .

4.3.3 Source revealed when query is posed, targets known

[083] In some embodiments the target locations of routing queries are known in advance, but the source is revealed only when a query is posed.

[084] In one embodiment, we use the graph G2 of Section 4.3.2 to compute a shortest ride.

[085] When a query (s, t) is posed, we determine walks from the location of s to each

$STOPSTATION_CLUSTER_SOURCE_c$.

In one embodiment, we use a shortest walk with duration that is at most a threshold, thereby generating a list of vehicle stops near the source location, or other embodiments as in Section 4.2.3. We also determine a shortest travel continuation from

$STOPSTATION_CLUSTER_SOURCE_c$

to

$TARGET_t$

in graph G2. In one embodiment, we pre-compute shortest path duration from each

$STOPSTATION_CLUSTER_SOURCE_c$

to each

$TARGET_t$,

and store the results. We look up these results from storage when a query is posed. In other embodiment, we use a graph shortest path algorithm in G2 to compute a duration when a query is posed.

[086] We find a cluster c that minimizes a sum of durations of a walk from s to c and a travel continuation from c to t . This minimum is a shortest travel duration from s to t .

[087] In other embodiment, we use a symmetric method when a target is revealed only when a query is posed. Then, instead of generating a list of vehicle stops near the source location, we generate a list of vehicle stops near the target location.

[088] In other embodiment, instead of using graph G2, we use graph G1'.

4.3.4 Target revealed when query is posed, sources known

[089] In some embodiments the source locations of routing queries are known in advance, but a target is revealed only when a query is posed.

[090] In one embodiment the graph G1 of Section 4.3.1 is used to compute a shortest ride. However, we need to compute prospect edges to the target. This computation is more involved than Section 4.3.2, because the target is unknown beforehand.

[091] We recall how choices were computed for each prospect edge in G0. For each clusters c and c' , let $choices(c, c')$ be these choices used to compute $P(c, c')$ for the edge from

$PROSPECT_CLUSTER_SOURCE_c$

to

$PROSPECT_CLUSTER_TARGET_c'$

in G0. It is possible that $choices(c, c')$ has just one choice (e.g., one bus, or one subway ride). The $choices(c, c')$ is defined even if the prospect edge was not added in G0 due to lack of a sufficient gain.

[092] Let the posed query be (s, t) , for a source

$SOURCE_s$

in the graph G1, and an arbitrary target location t that may be not represented in the graph.

[093] A shortest path from s to t may involve just one bus or only subways. In that case we need not consider prospect edges. We take the graph G1, and further extend it. We add vertex

$TARGET_t,$

and edges from

$STOPSTATION_CLUSTER_TARGET_c,$

for any c , to

$TARGET_t.$

Each of these edges is labelled Walk, and its weight is a walk duration. In one embodiment, any edge represents a shortest walk duration that is at most a threshold, or other embodiments as in Section 4.2.3. We compute a shortest path from

$SOURCE_s$

to

$TARGET_t$

in the resulting graph, and denote the path's length by $A(s, t)$. This length is a candidate for a shortest travel duration from s to t .

[094] There is other candidate. It is also possible that a shortest path involves more vehicles. In that case, there is a penultimate stop/station along the path. To cover this case, we compute prospect edges to t . The process is illustrated in FIG. 12. To simplify the illustration, the drawing depicts singleton prospect clusters (each has just one bus stop, or just one subway station).

[095] To compute prospect edges to t we start with a graph $G1$. We enumerate the parts of the journey from s to t that end at a penultimate stop/station. Specifically, we determine a shortest travel duration from

$SOURCE_s$

to

$PROSPECT_CLUSTER_SOURCE_c,$

for each c . We denote this duration by $shortest(s \rightarrow c)$. For example, in FIG. 12 the value 900 on the edge from

$SOURCE_s$

to

$SUBWAY_STATION_s_1$

denotes a shortest travel duration from

SOURCE_s

to

PROSPECT_CLUSTER_SOURCE_s1.

Note that this travel may pass along a prospect edge in the graph G1. In one embodiment, this duration can be pre-computed and stored before queries are posed, and looked up from storage upon a query.

[096] We determine how the journey can continue from each penultimate stop/station to the target t , using prospect edges and walks. For every

PROSPECT_CLUSTER_SOURCE_c,

we determine the choices of moving from c to t by first going to an intermediate

PROSPECT_CLUSTER_TARGET_c',

called $choices(c, c')$, and then following by a walk from c' to t . In one embodiment, we consider only shortest walks c' to t with duration that is at most a threshold, or other embodiments as in Section 4.2.3. For example, in FIG. 12 the $choices(b_1, b_0)$ are depicted on the edge from

BUS_STOP_b1

to

BUS_STOP_b0

— there are two choices: bus line e'' with wait duration uniform on $[0, 900]$ and travel duration 1600, and bus line e''' with wait duration uniform on $[0, 3600]$ and travel duration 1000. It takes 240 to continue by walk from

BUS_STOP_b0

to

TARGET_t.

[097] Because a rider located at c may pick any c' as a continuation, we combine at t the choices across all c' . This combination forms the choices for travel from c to t . For example, in FIG. 12 there is other edge from

$BUS_STOP_b_1$;

that edge goes to

$BUS_STOP_b_2$.

The $choices(b_1, b_2)$ depicted on that edge has just one choice: bus line e' with wait duration uniform on $[0, 300]$ and travel time 900. It takes 500 to continue by walk from

$BUS_STOP_b_2$

to

$TARGET_t$.

The combination of $choices(b_1, b_0)$ with $choices(b_1, b_2)$ yields three choices (bus lines e' , e'' and e'''). These are the choices of going from

$BUS_STOP_b_1$

to

$TARGET_t$.

An expected minimum travel time using these choices is 2636.

[098] We need to eliminate duplicate bus rides by the same bus line, like in Section 4.2.5. For example, in FIG. 12 a rider can depart from

$SUBWAY_STATION_s_1$

using the same bus line e'' , but going to two different locations:

$BUS_STOP_b_2$

and

SUBWAY_STATION_s0.

For any bus line at c , we retain only the choice for this bus line that has a lowest expected travel duration from c to t (eliminate any other choice for this bus line at c). For example, in FIG. 12 we eliminate the choice to

SUBWAY_STATION_s0

because it has a higher expectation. We compute an expected minimum travel duration, $P(c, t)$, among the remaining choices, similar to how we computed $P(c, c')$ in Section 4.2.5.

[099] A shortest path may pass any of the c , so we compute a minimum across c , and denote it $B(s, t)$

$$B(s, t) = \min_c \{ \text{shortest}(s \rightarrow c) + P(c, t) \}.$$

For example, in FIG. 12 the minimum $B(\text{SOURCE}_s, \text{TARGET}_t) = 2445$, which is $\min\{2636, 2445\}$, because it is more advantageous for the rider to travel to a penultimate

SUBWAY_STATION_s1,

rather than to a penultimate

BUS_STOP_b1.

[100] This quantity denotes a shortest travel duration from s to t that involves a penultimate vehicle. $B(s, t)$ is the other candidate for a shortest travel duration from s to t .

[101] Finally, a response to the query is a minimum of the two candidates: $\min\{A(s, t), B(s, t)\}$.

[102] For example, in FIG. 12 a response to the query is still 2445, because we cannot shorten travel by using just one vehicle that travels from

SOURCE_s

through

BUS_STOP_b0

to

TARGET_t,

because this travel duration is $A(SOURCE_s, TARGET_t) = 3000 + 240$.

[103] In other embodiment, instead of using graph G1, we use the graph G2'.

[104] In other embodiment, we use a symmetric method that computes prospect edges from a source, when the source is revealed only when a query is posed. Then, instead of considering a penultimate and a last stops before arriving at the target, we consider a first and a second stops after departing from the source.

4.3.5 Source and target revealed at query time

[105] When both the source and the target of a query are unknown beforehand, we select and combine the methods of previous sections. In one embodiment, we determine walks from the location of the source s to

$$STOPSTATION_CLUSTER_SOURCE_c,$$

for each c , and then travel from

$$STOPSTATION_CLUSTER_SOURCE_c$$

to target t (involving penultimate choices, or not). We respond with a minimum sum, selected across c . In one embodiment, we use a shortest walk with duration that is at most a threshold, or other embodiments as in Section 4.2.3.

4.4 Variants

[106] Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the embodiments. We present of few variants for illustration.

[107] In one embodiment, we use a more general notion of a prospect edge. When travel involves multiple vehicles and waits, a shortest path search in a graph may traverse multiple prospect edges, and these prospect edges along the path will together abstract a sequence of more than one wait and ride. To capture this multiplicity, in one embodiment, we use a more general notion of a depth- d prospect edge that abstracts a sequence of at most d waits&rides. For example, a path $c - walk1 - wait1 - bus1 - walk2 - wait2 - bus2 - walk3 - c'$ could

be abstracted as a depth-2 prospect edge from c to c' . In one embodiment, we add depth- d prospect edges for d larger than 1 to our graphs.

[108] In one embodiment, our method constructs routes given a departure time. For example, consider the case when the rider wishes to begin travel at 8 AM on a Tuesday. Here, a routing query specifies a departure time, in addition to the source and target locations of travel. In one embodiment, the source is a

$STOPSTATION_CLUSTER_SOURCE_s$,

and the target is a

$STOPSTATION_CLUSTER_TARGET_t$.

We modify the graph G_0 , see FIG. 1. Because here even the first ride may involve waiting, we remove the FirstWaitGetOn edges and the

$SUBWAY_FROM_s$

vertices, but add edges from each

$STOPSTATION_CLUSTER_SOURCE_c$

to

BUS_STOP_b

and

$SUBWAY_STATION_s$,

for any b and s in the cluster c . In one embodiment, we adopt the Dijkstra's shortest paths algorithm to use prospect edges: For each vertex

$PROSPECT_CLUSTER_SOURCE_c$,

we maintain a lowest known expected arrival time of the rider at the vertex, and use this time to condition the wait, walk and ride duration random variables to compute prospect edges to each

$PROSPECT_CLUSTER_TARGET_c'$.

810 Using thus computed edge weights, we update the lowest known expected arrival times at

811 $PROSPECT_CLUSTER_SOURCE_c'$.

812 In other embodiment, instead of maintaining or updating a lowest known expected arrival
813 time at each

814 $PROSPECT_CLUSTER_SOURCE_c$

815 or at each

816 $PROSPECT_CLUSTER_TARGET_c$,

817 we maintain or update a probability distribution of arrival time. In other embodiment, we
818 adopt other shortest paths algorithms, for example the A* (A star) search algorithm in a
819 similar fashion. In other embodiment, for example when a departure time is “now/soon”,
820 the conditional random variables are computed using the state of the transportation system
821 at the time of the query, to provide more accurate distributions of wait and ride durations.

822 [109] In one embodiment, our method constructs routes given an arrival deadline. For
823 example, consider the case when the rider wishes to arrive at the target before 9 AM on a
824 Tuesday. This is equivalent to departure from the target at 9 AM, but going back in time
825 and space. This can be simply abstracted through an appropriately reversed construction of
826 any of our graphs (buses and subways travel in reverse time and space).

827 [110] In one embodiment, we determine prospect travel that meets a desired probability
828 p of arrival before a deadline. When considering a prospect edge from c to c' , we use a
829 random variable A denoting an arrival time of the rider at c . Then, given the k random
830 variables $T_1, \dots, T_k$ of travel duration from c to c' using choices, we determine a distribution
831 of arrival time at c' using the prospect travel, $\min(A + T_1, \dots, A + T_k)$. Then we determine
832 up to which time t this distribution has the mass that is the desired probability p .

833 [111] In one embodiment, we report the vehicles along a shortest path, or times of arrival/
834 departure for each point along the path. This information can be simply read off the path
835 in the graph and the choices of prospect edges along the path.

836 [112] In one embodiment, we answer routing queries on computing devices with limited
837 storage and restricted communication with a backend server. For example, this can happen

on a mobile phone for a user concerned about privacy. In that case, we use an appropriately small number of clusters in graph G0. Similar techniques can be used in our other graphs.

[113] In one embodiment, we impose requirements on a routing answer, including a maximum walk duration, a maximum number of transfers, a maximum wait duration, a restriction to specific types of vehicles (e.g., use only express bus and subway). Our invention realizes these requirements by an appropriate modification of graphs and a shortest paths algorithm on the graphs.

[114] In one embodiment, our method is applied to an imperfect graph. For example, the weight of an edge WaitGetOn could inexactly reflect an expected wait duration for a subway, perhaps because we estimated the duration incorrectly, or there could be vertices and edges for a bus that does not exist in the metropolitan area, perhaps because the bus route was just cancelled by the city government while our method was not yet able to notice the cancellation, or we sampled the expectation of the minimum of choices with a large error, or used an approximate mathematical formula/algorithm. These are just a few non-exhaustive examples of imperfectness. In any case, our method can still be applied. It will simply produce routes with some error.

[115] In one embodiment, we remove unnecessary vertices and edges from a graph. For example, we collapse “pass through” vertices

$$SUBWAY_AVG_FROM_TO_s'_s''$$

in G0 by fusing the incoming edge and the outgoing edge.

[116] In one embodiment, the steps of our method are applied in other order. For example, when constructing graph G0, we can reverse the order described in Sections 4.2.1 and 4.2.2: first add vertices and edges of the non-fixed timetable vehicles, and then add vertices and edges of fixed timetable vehicles.

[117] In one embodiment, we parallelize the method. For example, instead of computing the prospect edges from each

$$PROSPECT_CLUSTER_SOURCE_c$$

in turn, we can consider any two c_1 and c_2 , and compute the prospect edges from

$$PROSPECT_CLUSTER_SOURCE_c_1$$

in parallel with computing the prospect edges from

*PROSPECT_CLUSTER_SOURCE_* c_2 .

5 Computer system

[118] One of the embodiments of the invention is a computer system that answers routing queries.

[119] In one embodiment, the system answers queries for a shortest route between locations, given a departure time: any query is in the form (source, target, minuteOfDay). An answer is in the form of a route with durations and choices. An illustration of the embodiment is in FIG. 13.

[120] We use the term “module” in our description. It is known in the art that the term means a computer (sub)system that provides some specific functionality. Our choice of partitioning the computer system into the specific modules is exemplary, not mandatory. Those of ordinary skill in the art will notice that the system can be organized into modules in other manner without departing from the scope of the invention.

[121] One module of the system (1202) reads information about the metropolitan transportation system from a plurality of data sources (1201). The module determines which vehicles, routes, or their parts, are consider fixed timetable, and which non-fixed timetable. The module computes routes for fixed timetable vehicles. The module also computes distributions of wait and ride durations conditioned on time for non-fixed timetable vehicles.

[122] The output is passed to a module (1204) that computes prospect edges. That module queries information about walks from a plurality of data sources (1203). For selected prospect clusters c and c' and arrival times of the rider at c , the module computes the weight $P(c, c')$ of the prospect edge and the choices, using random variables conditioned on the rider arrival time at c . The results are stored in storage (1205).

[123] The modules (1202) and (1204) operate continuously. As a result, the system maintains a fresh model of the transportation system.

[124] In the meantime, other module (1206) pre-computes shortest paths. The module constructs graphs that link locations at times by reading prospect edges from (1205) and

895 non-prospect edges from (1202). Shortest paths algorithms are applied to the graphs to
 896 compute paths for selected queries in the form (stop/station cluster source, stop/station
 897 cluster target, minuteOfDay). The results are stored, so that a result can be looked up from
 898 storage (1207) when needed later.

899 [125] Concurrently, the query answering module (1208) answers queries. When a query
 900 (source, target, minuteOfDay) arrives (1209), the module computes a shortest path following
 901 Section 4. The module contacts (1203) to determine walks between the source and the target,
 902 and the stop/station clusters. The module looks up relevant pre-computed shortest paths
 903 from (1207). When one is needed but not available yet, the module requests a shortest path
 904 from module (1206), and may store the resulting shortest path in storage (1207) for future
 905 use. The module (1208) also looks up choices and times from (1205). These walks, shortest
 906 paths and choices are combined to generate an answer to the query (1210).

907 [126] Aspects of the invention may take form of a hardware embodiment, a software
 908 embodiment, or a combination of the two. Steps of the invention, including blocks of any
 909 flowchart, may be executed out of order, partially concurrently or served from a cache, de-
 910 pending on functionality and optimization. Aspects may take form of a sequential system,
 911 or parallel/distributed system, where each component embodies some aspect, possibly re-
 912 dundantly with other components, and components may communicate, for example using a
 913 network of any kind. A computer program carrying out operations for aspects of the inven-
 914 tion may be written in any programming language, including C++, Java or JavaScript. Any
 915 program may execute on an arbitrary hardware platform, including a Central Processing
 916 Unit (CPU), and a Graphics Processing Unit (GPU), and associated memory and storage
 917 devices. A program may execute aspects of the invention on one or more software platforms,
 918 including, but not limited to: a smartphone running Android or iOS operating systems, or
 919 a web browser, including Firefox, Chrome, Internet Explorer, or Safari.

920 **6 Computer service product**

921 [127] One of the embodiments of the invention is a service product available to users
 922 through a user-facing device, such as a smartphone application or a webpage. It will be
 923 obvious to anyone of ordinary skill in the art that the invention is not limited to these

924 devices. It will also be obvious that the presentation of the service in our drawings can
 925 be modified (including rearranging, resizing, changing colors, shape, adding or removing
 926 components) without departing from the scope of the invention.

927 [128] In one embodiment, the service is accessed through a smartphone application. A
 928 user specifies a departure time, and a source and a target, by interacting with the User
 929 Interface of the application on the smartphone. The service then generates a route, and
 930 renders a representation of the route on the smartphone. FIG. 14 illustrates an example
 931 result for a query from A to L departing at 8 AM.

932 [129] In one embodiment, the service reports which choice yields a shortest travel duration
 933 currently. In one embodiment, the system highlights this faster choice (illustrated by 1401
 934 on the route), or shows the travel duration by the choice, or depicts a current wait duration
 935 (illustrated by 1402 near D). In one embodiment, this faster choice is computed given the
 936 current positions of the vehicles. In one embodiment, this report is rendered when the user
 937 is currently near the location of the choice; for example, when the user is about to depart
 938 from A, the service may render that in the current conditions, it is faster to get to F via D
 939 and E, rather than via B and C.

940 [130] In one embodiment, the service depicts a current wait duration for each choice from
 941 among the choices at a location (illustrated by 1403 near F), or an expected ride duration
 942 for each choice (illustrated by 1404). This may help the user decide by themselves which
 943 choice to take, even if not optimal.

944 [131] In one embodiment, the service reports one expected wait duration for all the choices
 945 at a location. The duration is an expected wait duration assuming the user will board the
 946 choice that achieves an expected minimum travel duration (illustrated by 1405 near H). In
 947 one embodiment, this report is rendered when vehicle positions are uncertain, for example
 948 for a segment of the route far down the road compared to the current position of the user.
 949 This informs the user how long they will idle at a stop/station waiting for a vehicle.

950 [132] In one embodiment, the service reports a duration of prospect travel between two
 951 locations (illustrated by 1406 on the route). This is the value denoted $P(c, c')$ in Section 4.

952 [133] In one embodiment, the service responds to the user with at least one of:

- 953 1. the source location rendered on a map;

- 954 2. the target location rendered on a map;
- 955 3. a location of any stop along a route rendered on a map;
- 956 4. a sequence of locations along a route rendered on a map;
- 957 5. a name, an address, or an identifier of any of: the source location, the target location,
958 or any stop along a route;
- 959 6. a departure time;
- 960 7. a departure time range;
- 961 8. an arrival time;
- 962 9. an arrival time range;
- 963 10. a probability of arriving before a deadline;
- 964 11. a sequence of locations along two or more choices that travel between two locations
965 rendered on a map;
- 966 12. directions for a walk component in any choice;
- 967 13. a location or a duration of a wait component in any choice;
- 968 14. directions for a ride component in any choice;
- 969 15. an expected minimum wait duration among at least two choices;
- 970 16. a current minimum wait duration among at least two choices;
- 971 17. an expected travel duration for any component of a choice, or a choice; or an expected
972 minimum travel duration among at least two choices;
- 973 18. a current travel duration for any component in a choice, or a choice; or a minimum
974 travel duration among at least two choices;
- 975 19. an expected departure time or an expected arrival time for : any component in a
976 choice, a choice, or a minimum among at least two choices;

- 977 20. a current departure time or a current arrival time for : any component in a choice, a
978 choice, or a minimum among at least two choices;
- 979 21. a name or an identifier of any vehicle in any choice;
- 980 22. a name, an address, or an identifier of any stop of any vehicle in any choice;
- 981 23. a current location of any vehicle in any choice; or
- 982 24. a rendering of which choice, from among two or more choices, is fastest given current
983 locations of vehicles.

984 7 Claims

985 [134] Those skilled in the art shall notice that various modifications may be made, and
986 substitutions may be made with essentially equivalents, without departing from the scope of
987 the present invention. Besides, a specific situation may be adapted to the teachings of the
988 invention without departing from its scope. Therefore, despite the fact that the invention
989 has been described with reference to the disclosed embodiments, the invention shall not be
990 restricted to these embodiments. Rather, the invention will include all embodiments that
991 fall within the scope of the appended claims.

CLAIMS

1. A method for providing a route or a route duration for a journey from a source location to a target location, the method comprising:
 - (a) receiving a request comprising the source location and the target location;
 - (b) determining two or more travel paths from the source location to the target location;
 - (c) determining a travel duration for each of the travel paths, wherein the travel duration is defined as a mathematical random variable that is non-trivial;
 - (d) generating a route or a route duration using a mathematical random variable M that is a minimum of the travel duration mathematical random variables, wherein the using comprises at least one of:
 - i. evaluating an expected value of the mathematical random variable M , or
 - ii. computing a probability of arrival time $Pr[T + M \leq A]$ for at least one threshold T on a departure time and at least one threshold A on an arrival time; and
 - (e) responding to the request with information comprising the route or the route duration.
2. The method of claim 1, wherein the request further comprises a time of departure from the source location.
3. The method of claim 1, wherein at least one of the travel paths includes at least one ride by a vehicle.
4. The method of claim 3, wherein the vehicle follows a non-fixed timetable.
5. The method of claim 3, wherein the route is constrained by at least one of:
 - (a) a vehicle type;
 - (b) a vehicle stop type; or
 - (c) a threshold on the number of vehicle transfers.

- 1018 6. The method of claim 3, wherein a ride duration of the vehicle is a mathematical
1019 random variable conditioned on a departure time from a location where the vehicle is
1020 boarded.
- 1021 7. The method of claim 1, wherein at least one of the travel paths includes at least one
1022 wait for a vehicle.
- 1023 8. The method of claim 7, wherein the route is constrained by a threshold on a wait
1024 duration.
- 1025 9. The method of claim 7, wherein a wait duration for the vehicle is a mathematical
1026 random variable conditioned on an arrival time at a location where the vehicle is
1027 boarded.
- 1028 10. The method of claim 1, wherein at least one of the travel paths includes at least one
1029 walk.
- 1030 11. The method of claim 10, wherein the route is constrained by a threshold on a walk
1031 duration.
- 1032 12. The method of claim 1, wherein a probability distribution of at least one of the travel
1033 duration mathematical random variables is determined from at least one of:
- 1034 (a) historical data about an arrival or a departure time of a vehicle from a stop;
1035 (b) historical data about a duration of travel of a vehicle between stops;
1036 (c) passing interval of a vehicle through a stop; or
1037 (d) current geographic location of a vehicle.
- 1038 13. The method of claim 1, wherein at least one of the travel duration mathematical
1039 random variables is conditioned on an arrival time at the source location.
- 1040 14. The method of claim 1, wherein at least one of the travel duration mathematical
1041 random variables is uniformly distributed on an interval between a minimum travel
1042 duration and a maximum travel duration.

- 1043 15. The method of claim 1, wherein at least one of the travel duration mathematical
1044 random variables is conditioned on a source location arrival time that falls within a
1045 specific time window.
- 1046 16. The method of claim 1, wherein portions of two or more of the travel paths are over-
1047 lapping.
- 1048 17. The method of claim 1, wherein the evaluating an expected value of the mathematical
1049 random variable M comprises at least one of:
- 1050 (a) sampling from a distribution of the mathematical random variable M;
 - 1051 (b) evaluating a mathematical formula for an expected value of the mathematical
1052 random variable M;
 - 1053 (c) computing an approximate integral over a distribution of the mathematical ran-
1054 dom variable M; or
 - 1055 (d) executing an approximation algorithm for an expected value of the mathematical
1056 random variable M.
- 1057 18. The method of claim 1, wherein the generating a route or a route duration comprises:
- 1058 (a) determining two or more first type travel duration mathematical random variables
1059 that are correlated;
 - 1060 (b) determining one or more second type travel duration mathematical random vari-
1061 ables that are independent; and
 - 1062 (c) computing an expected minimum of : one first type random variable and all
1063 second type random variables.
- 1064 19. The method of claim 1, wherein the route is constrained by at least one of:
- 1065 (a) a type of object being routed; or
 - 1066 (b) a threshold on the monetary cost of travel.
- 1067 20. The method of claim 1,
1068 wherein the request further comprises:

(a) an arrival deadline, and a desired probability of arriving at the target location before the arrival deadline;

wherein the computing a probability of arrival time $Pr[T + M \leq A]$ for at least one threshold T on a departure time and at least one threshold A on an arrival time comprises:

(b) setting the threshold A to the arrival deadline, and determining a threshold T, that is the departure time from the source location, such that the probability $Pr[T + M \leq A]$ is equal to the desired probability of arriving at the target location before the arrival deadline.

21. The method of claim 1,

further comprising: before the receiving a request,

(a) pre-computing at least one of the following items:

- i. a travel duration mathematical random variable for a travel path;
- ii. an expected minimum of at least two travel duration mathematical random variables;
- iii. a probability distribution of a minimum of at least two travel duration mathematical random variables; or
- iv. a route or a route duration between a pair of locations; and

(b) storing the pre-computed item in a database;

wherein the generating a route or a route duration comprises:

(c) retrieving from the database the pre-computed item.

22. The method of claim 3,

wherein the determining of two or more travel paths comprises:

(a) determining at least one vehicle stop location within a threshold distance from the source location, and at least one vehicle stop location within the threshold distance from the target location, thereby generating two lists of vehicle stops near the source and target locations;

wherein the generating a route or a route duration comprises:

- (b) computing the route duration using at least one travel duration mathematical random variable of at least one pair of vehicle stops from the lists; and
- (c) determining walk durations between the vehicle stops and the source and the target locations.

23. A method for providing a route or a route duration for a journey from a source location to a target location, the method comprising:

- (a) receiving information about a plurality of stops of a vehicle and a vehicle travel time information, for a plurality of vehicles;
- (b) creating a graph comprising a plurality of graph vertices and a plurality of graph edges, including at least two graph vertices that represent vehicle stops and at least two graph edges that represent vehicle travel durations;
- (c) creating at least one graph prospect edge from an origin vertex to a destination vertex, each prospect edge representing a travel duration, and comprising:
 - i. determining two or more travel paths from the location of the origin vertex of the graph prospect edge to the location of the destination vertex of the graph prospect edge;
 - ii. determining a travel duration for each of the travel paths, wherein the travel duration is defined as a mathematical random variable that is non-trivial;
 - iii. computing a graph prospect edge travel duration using a mathematical random variable M that is a minimum of the travel duration mathematical random variables, wherein the using comprises at least one of:
 - A. evaluating an expected value of the mathematical random variable M , or
 - B. computing a probability of arrival time $Pr[T + M \leq A]$ for at least one threshold T on a departure time and at least one threshold A on an arrival time; and
- (d) receiving at least one request for a route or a route duration from a source location to a target location;

1124 (e) generating a route or a route duration using at least one graph path between
 1125 arbitrary two vertices, wherein the at least one graph path includes the graph
 1126 prospect edge; and

1127 (f) responding to the request with information comprising the route or the route
 1128 duration.

1129 24. The method of claim 23, wherein

1130 (a) at least one vehicle of the plurality of vehicles follows a non-fixed timetable; and

1131 (b) the plurality of graph edges includes at least one [RideSame] graph edge repre-
 1132 senting a travel duration between two stops of the non-fixed timetable vehicle.

1133 25. The method of claim 24, wherein the [RideSame] graph edge represents an average
 1134 travel duration during a specific time window.

1135 26. The method of claim 23, wherein

1136 (a) at least one vehicle of the plurality of vehicles follows a fixed timetable; and

1137 (b) the plurality of graph edges includes at least one [RideManyGetOff] graph edge
 1138 representing a travel duration between two stops of at least one fixed timetable
 1139 vehicle.

1140 27. The method of claim 23, wherein the plurality of graph edges includes at least one
 1141 [WaitGetOn] graph edge representing a wait duration for a vehicle before boarding the
 1142 vehicle at a stop.

1143 28. The method of claim 27, wherein the [WaitGetOn] graph edge represents a wait
 1144 duration equal to half of an average inter-arrival time of a vehicle at a stop during a
 1145 specific time window.

1146 29. The method of claim 23, wherein the plurality of graph edges includes at least one
 1147 [Walk] graph edge representing a walk duration between vertex locations.

1148 30. The method of claim 23, wherein the plurality of graph edges includes at least one of:

1149 (a) a [GetOff] graph edge representing getting off from a vehicle onto a stop;

- (b) a [FirstWaitGetOn] graph edge representing boarding a vehicle without waiting;
 or
 (c) a [Zero] graph edge representing a travel between vertices that lasts less than a threshold duration.

31. The method of claim 23, wherein

- (a) the plurality of graph vertices includes at least one vertex representing a location other than any vehicle stop; and
 (b) the plurality of graph edges includes at least one [Walk] graph edge representing a walk duration between the vertex other than any vehicle stop and any other graph vertex.

32. The method of claim 23, wherein

- (a) the plurality of graph vertices includes a cluster vertex representing at least two graph vertices clustered based on a geographical proximity; and
 (b) the plurality of graph edges includes at least one edge representing a travel duration between the cluster vertex and any other graph vertex.

33. The method of claim 23, wherein the creating at least one graph prospect edge comprises creating a graph prospect edge to each of one or more destination vertices from an origin vertex comprising:

- (a) traversing the graph forward from the origin vertex;
 (b) creating a plurality of graph paths that end at any of the destination vertices; and
 (c) identifying the graph paths that lead to a destination vertex, for each of the destination vertices.

34. The method of claim 23, wherein the creating at least one graph prospect edge comprises creating a graph prospect edge from each of one or more origin vertices to a destination vertex comprising:

- (a) traversing the graph backward from the destination vertex;

- 1177 (b) creating a plurality of graph paths that begin at any of the origin vertices; and
- 1178 (c) identifying the graph paths that lead from an origin vertex, for each of the origin
- 1179 vertices.
- 1180 35. The method of claim 23, wherein at least one of the travel paths includes a travel by
- 1181 two or more vehicles or by two or more walks.
- 1182 36. The method of claim 23, wherein the creating at least one graph prospect edge is
- 1183 performed with a condition that a graph prospect edge for the two or more travel
- 1184 paths is not created when the $k \geq 2$ travel duration mathematical random variables
- 1185 $T_1, \dots, T_k$ satisfy a condition that a minimum of expected values $\min_{1 \leq i \leq k} \mathbb{E}[T_i]$ is
- 1186 within a threshold from a value of expected minimum $\mathbb{E}[\min_{1 \leq i \leq k} T_i]$.
- 1187 37. The method of claim 23, wherein the generating a route or a route duration comprises
- 1188 applying a Dijkstra's shortest paths algorithm or an A* (A star) search algorithm to
- 1189 the graph.
- 1190 38. The method of claim 23, wherein the request further comprises a time of departure
- 1191 from the source location.
- 1192 39. The method of claim 23, wherein at least one of the travel paths includes at least one
- 1193 ride by a vehicle.
- 1194 40. The method of claim 39, wherein the vehicle follows a non-fixed timetable.
- 1195 41. The method of claim 39, wherein the route is constrained by at least one of:
- 1196 (a) a vehicle type;
- 1197 (b) a vehicle stop type; or
- 1198 (c) a threshold on the number of vehicle transfers.
- 1199 42. The method of claim 39, wherein a ride duration of the vehicle is a mathematical
- 1200 random variable conditioned on a departure time from a location where the vehicle is
- 1201 boarded.

- 1202 43. The method of claim 23, wherein at least one of the travel paths includes at least one
1203 wait for a vehicle.
- 1204 44. The method of claim 43, wherein the route is constrained by a threshold on a wait
1205 duration.
- 1206 45. The method of claim 43, wherein a wait duration for the vehicle is a mathematical
1207 random variable conditioned on a probability distribution of arrival time at a location
1208 where the vehicle is boarded.
- 1209 46. The method of claim 23, wherein at least one of the travel paths includes at least one
1210 walk.
- 1211 47. The method of claim 46, wherein the route is constrained by a threshold on a walk
1212 duration.
- 1213 48. The method of claim 23, wherein a probability distribution of at least one of the travel
1214 duration mathematical random variables is determined from at least one of:
- 1215 (a) historical data about an arrival or a departure time of a vehicle from a stop;
1216 (b) historical data about a duration of travel of a vehicle between stops;
1217 (c) passing interval of a vehicle through a stop; or
1218 (d) current geographic location of a vehicle.
- 1219 49. The method of claim 23, wherein for at least one graph prospect edge, at least one
1220 of the travel duration mathematical random variables is conditioned on a probability
1221 distribution of arrival time at a location of the origin vertex of the prospect edge.
- 1222 50. The method of claim 23, wherein at least one of the travel duration mathematical
1223 random variables is uniformly distributed on an interval between a minimum travel
1224 duration and a maximum travel duration.
- 1225 51. The method of claim 23, wherein for at least one graph prospect edge, at least one of
1226 the travel duration mathematical random variables is conditioned on an arrival time
1227 at a location of the origin vertex of the prospect edge that falls within a specific time
1228 window.

1229 52. The method of claim 23, wherein portions of two or more of the travel paths are
1230 overlapping.

1231 53. The method of claim 23, wherein the evaluating an expected value of the mathematical
1232 random variable M comprises at least one of:

1233 (a) sampling from a distribution of the mathematical random variable M;

1234 (b) evaluating a mathematical formula for an expected value of the mathematical
1235 random variable M;

1236 (c) computing an approximate integral over a distribution of the mathematical ran-
1237 dom variable M; or

1238 (d) executing an approximation algorithm for an expected value of the mathematical
1239 random variable M.

1240 54. The method of claim 23, wherein the computing a graph prospect edge travel duration
1241 comprises:

1242 (a) determining two or more first type travel duration mathematical random variables
1243 that are correlated;

1244 (b) determining one or more second type travel duration mathematical random vari-
1245 ables that are independent; and

1246 (c) computing an expected minimum of : one first type random variable and all
1247 second type random variables.

1248 55. The method of claim 23, wherein the route is constrained by at least one of:

1249 (a) a type of object being routed; or

1250 (b) a threshold on the monetary cost of travel.

1251 56. The method of claim 23,

1252 wherein the request further comprises:

1253 (a) an arrival deadline, and a desired probability of arriving at the target location
1254 before the arrival deadline;

wherein the computing a probability of arrival time $Pr[T + M \leq A]$ for at least one threshold T on a departure time and at least one threshold A on an arrival time comprises:

- (b) setting the threshold A to the arrival deadline, and determining a threshold T, that is a mathematical random variable of the departure time from the location of the origin vertex of the graph prospect edge, such that the probability $Pr[T + M \leq A]$ is equal to the desired probability of arriving at the target location before the arrival deadline.

57. The method of claim 23,

further comprising: before the receiving at least one request,

- (a) pre-computing at least one of the following items:
 - i. a travel duration mathematical random variable between a pair of vertices;
 - ii. an expected minimum of at least two travel duration mathematical random variables;
 - iii. a probability distribution of a minimum of at least two travel duration mathematical random variables; or
 - iv. a graph path or a travel duration between a pair of vertices; and

- (b) storing the pre-computed item in a database;

wherein the generating a route or a route duration comprises:

- (c) retrieving from the database the pre-computed item.

58. The method of claim 23,

further comprising:

- (a) before the receiving at least one request, pre-computing a route or a route duration for at least one pair of vertices, and storing the pre-computed route or the pre-computed route duration in a database,

wherein the generating a route or a route duration comprises:

(b) determining at least one vertex whose location is within a threshold distance from the source location, and at least one vertex whose location is within a threshold distance from the target location, thereby generating two lists of vertices near the source and target locations; and

(c) for at least one pairwise combination of vertices on the two lists, retrieving from the database at least one route or route duration.

59. The method of claim 23, wherein the generating a route or a route duration comprises:

(a) determining at least one initial part of travel involving travel from the source location to a second vehicle stop, wherein the initial part of travel is determined by:

i. generating two or more travel paths from the source location, each travel path including at least one vehicle ride from a first vehicle stop that is within a threshold distance from the source location to the second vehicle stop, where the first vehicle stop is an arbitrary stop of any vehicle, and the second vehicle stop is an arbitrary stop of any vehicle; and

ii. determining a travel duration for each of the travel paths, wherein the travel duration is defined as a mathematical random variable; and

(b) computing a duration of at least one initial part of travel using a mathematical random variable that is a minimum of the travel duration mathematical random variables.

60. The method of claim 23, wherein the generating a route or a route duration comprises:

(a) determining at least one final part of travel involving travel from a penultimate vehicle stop to the target location, wherein the final part of travel is determined by:

i. generating two or more travel paths to the target location, each travel path including at least one vehicle ride from the penultimate vehicle stop to a last vehicle stop that is within a threshold distance from the target location, where the penultimate vehicle stop is an arbitrary stop of any vehicle, and the last vehicle stop is an arbitrary stop of any vehicle; and

- 1310 ii. determining a travel duration for each of the travel paths, wherein the travel
1311 duration is defined as a mathematical random variable; and
- 1312 (b) computing a duration of at least one final part of travel using a mathematical
1313 random variable that is a minimum of the travel duration mathematical random
1314 variables.
- 1315 61. A computer system for providing a route or a route duration for a journey from a
1316 source location to a target location, the computer system comprising:
- 1317 (a) one or more processors;
- 1318 (b) memory storing one or more programs for execution by one or more processors;
1319 and
- 1320 (c) the one or more programs comprising instructions to be executed by the one or
1321 more processors so as to perform the method of any of claims 1 to 60.
- 1322 62. The computer system of claim 61, wherein the information comprising the route or the
1323 route duration further comprises at least one of:
- 1324 (a) the source location rendered on a map;
- 1325 (b) the target location rendered on a map;
- 1326 (c) a location of any stop along a route rendered on a map;
- 1327 (d) a sequence of locations along a route rendered on a map;
- 1328 (e) a name, an address, or an identifier of any of: the source location, the target
1329 location, or any stop along a route;
- 1330 (f) a departure time;
- 1331 (g) a departure time range;
- 1332 (h) an arrival time;
- 1333 (i) an arrival time range;
- 1334 (j) a probability of arriving before a deadline;
- 1335 (k) a sequence of locations along two or more travel paths between two locations
1336 rendered on a map;

- 1337 (l) directions for a walk component in any travel path;
- 1338 (m) a location or a duration of a wait component in any travel path;
- 1339 (n) directions for a ride component in any travel path;
- 1340 (o) an expected minimum wait duration among at least two travel paths;
- 1341 (p) a current minimum wait duration among at least two travel paths;
- 1342 (q) an expected travel duration for any component of a travel path, or a travel path;
- 1343 or an expected minimum travel duration among at least two travel paths;
- 1344 (r) a current travel duration for any component in a travel path, or a travel path; or
- 1345 a minimum travel duration among at least two travel paths;
- 1346 (s) an expected departure time or an expected arrival time for : any component in a
- 1347 travel path, a travel path, or a minimum among at least two travel paths;
- 1348 (t) a current departure time or a current arrival time for : any component in a travel
- 1349 path, a travel path, or a minimum among at least two travel paths;
- 1350 (u) a name or an identifier of any vehicle in any travel path;
- 1351 (v) a name, an address, or an identifier of any stop of any vehicle in any travel path;
- 1352 (w) a current location of any vehicle in any travel path; or
- 1353 (x) a rendering of which travel path, from among two or more travel paths, is fastest
- 1354 given current locations of vehicles.

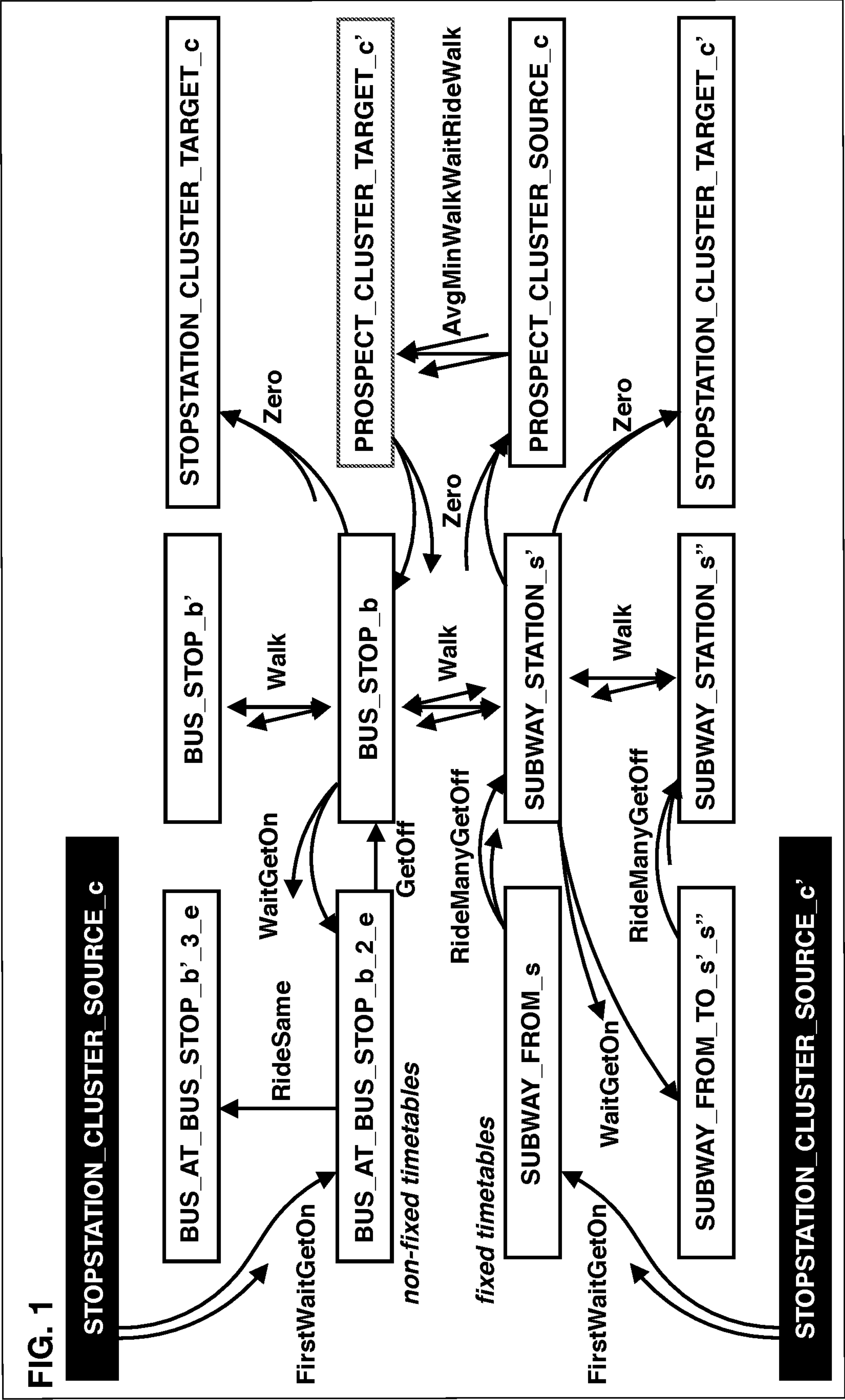
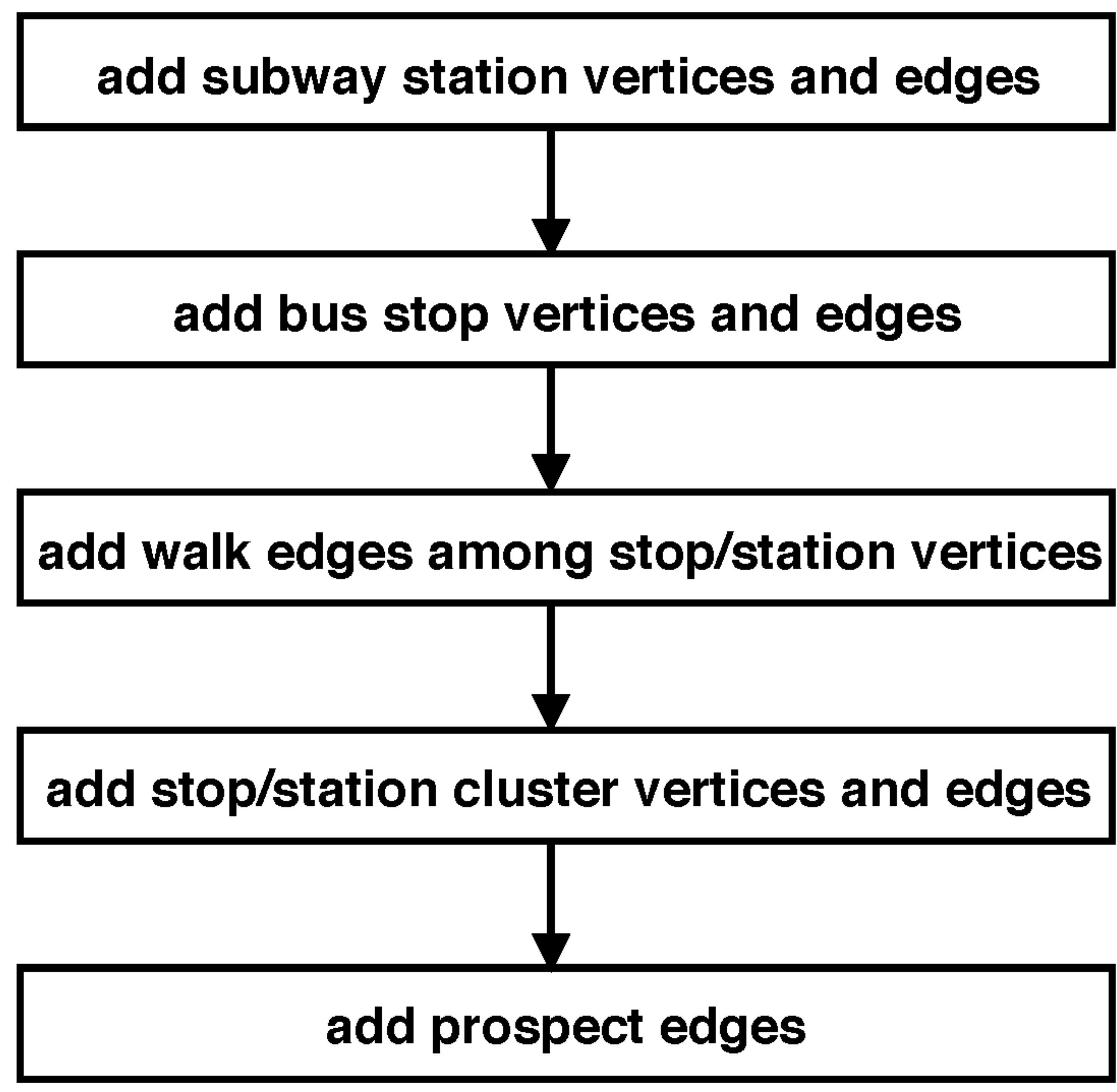
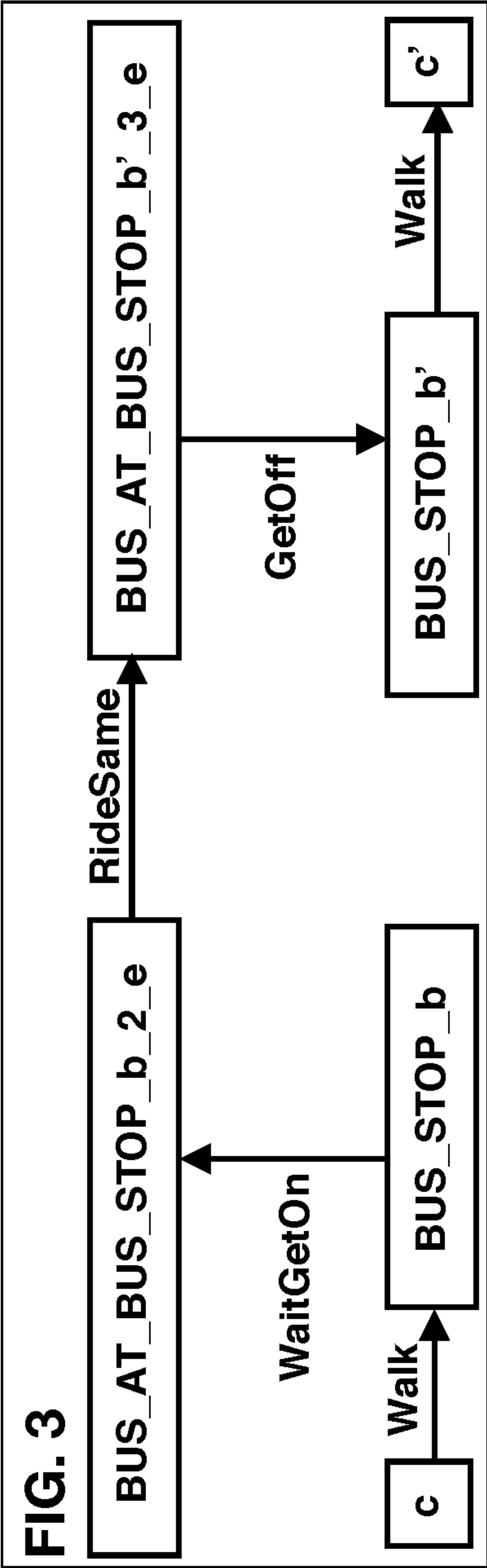
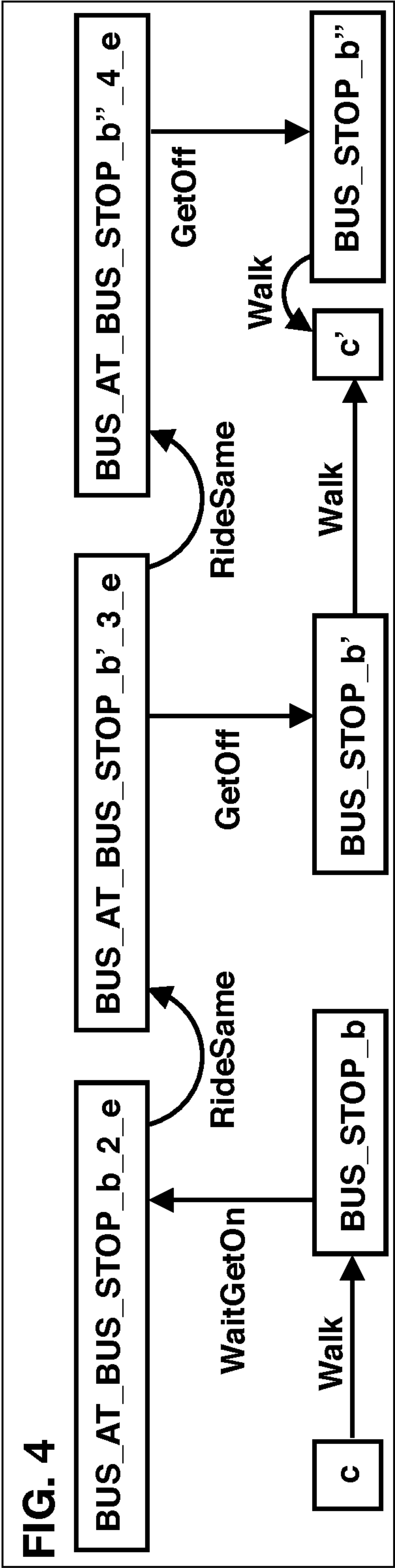
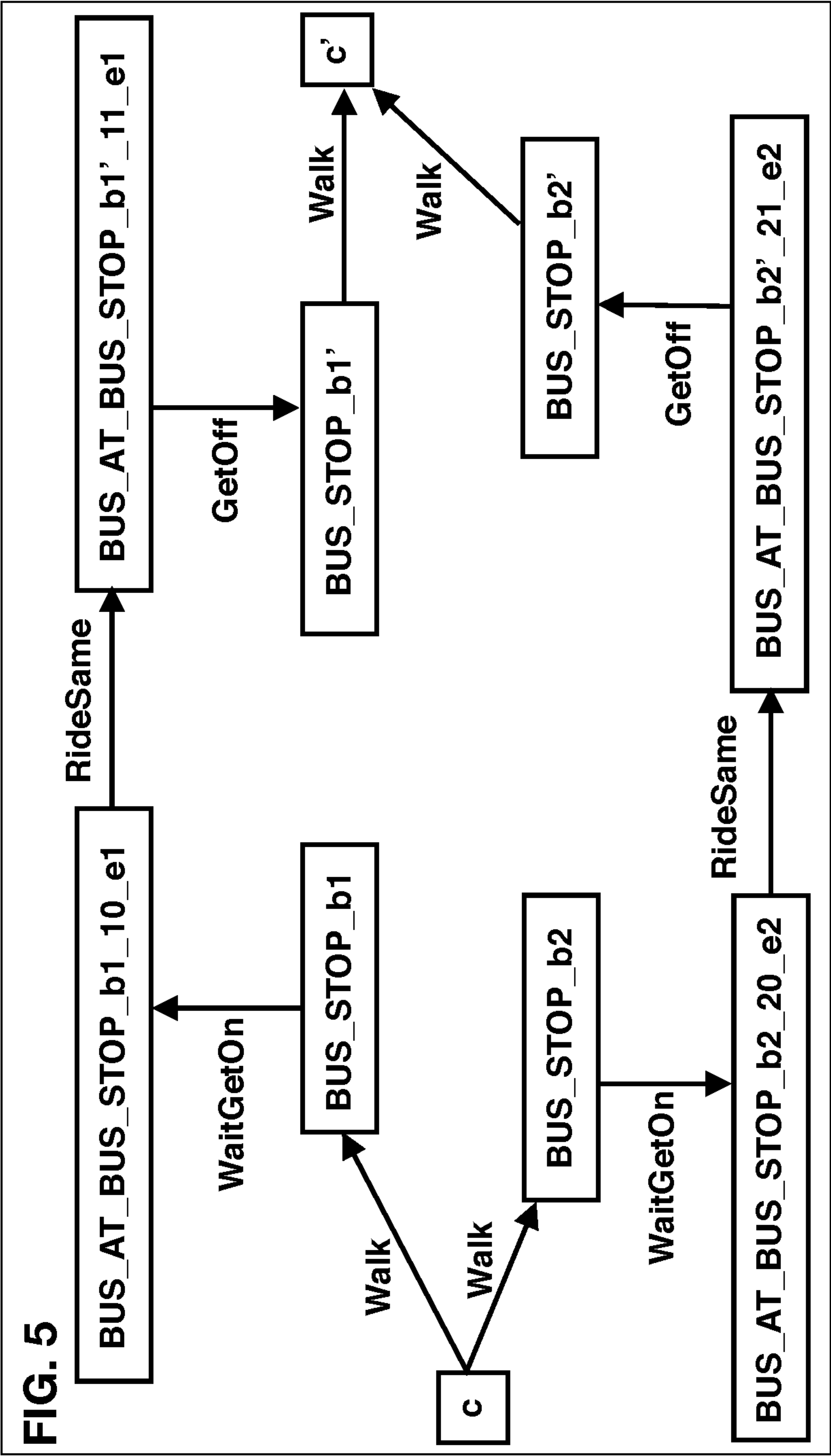
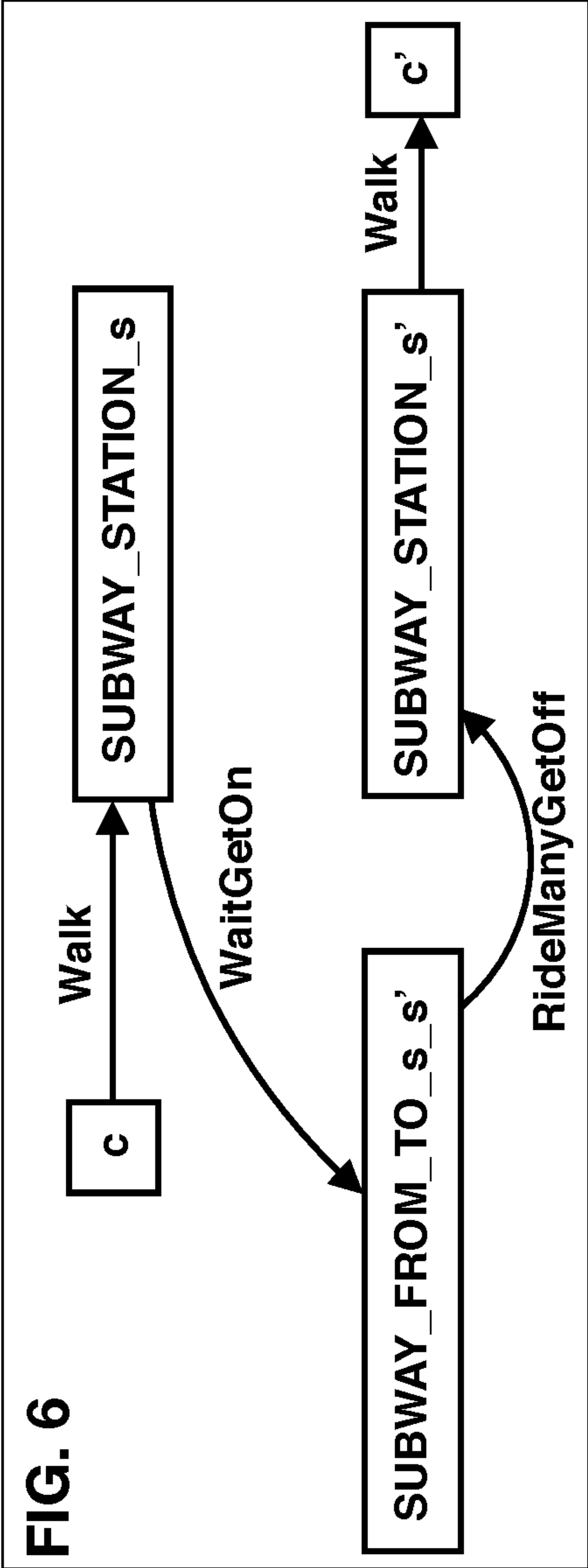


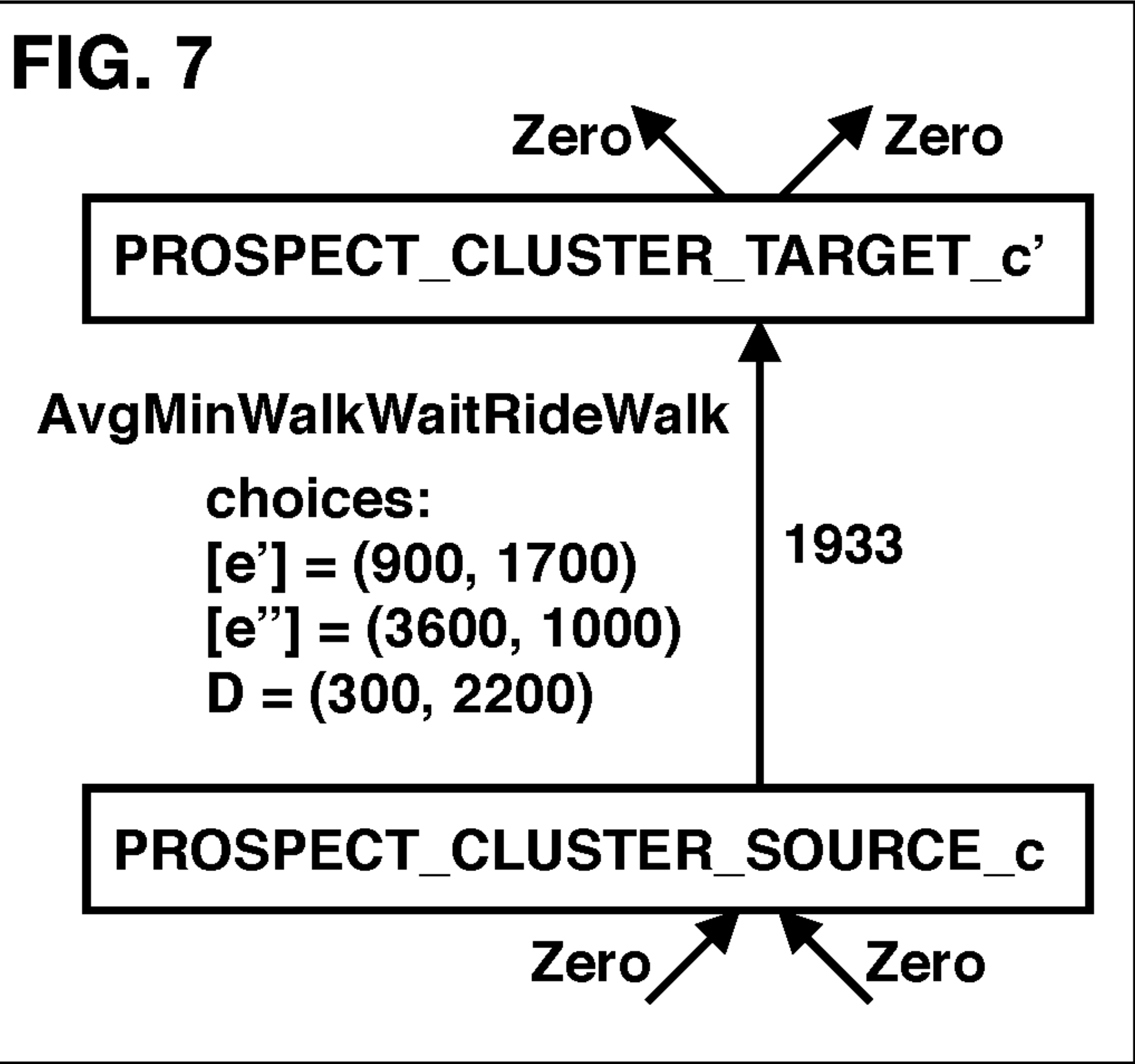
FIG. 2











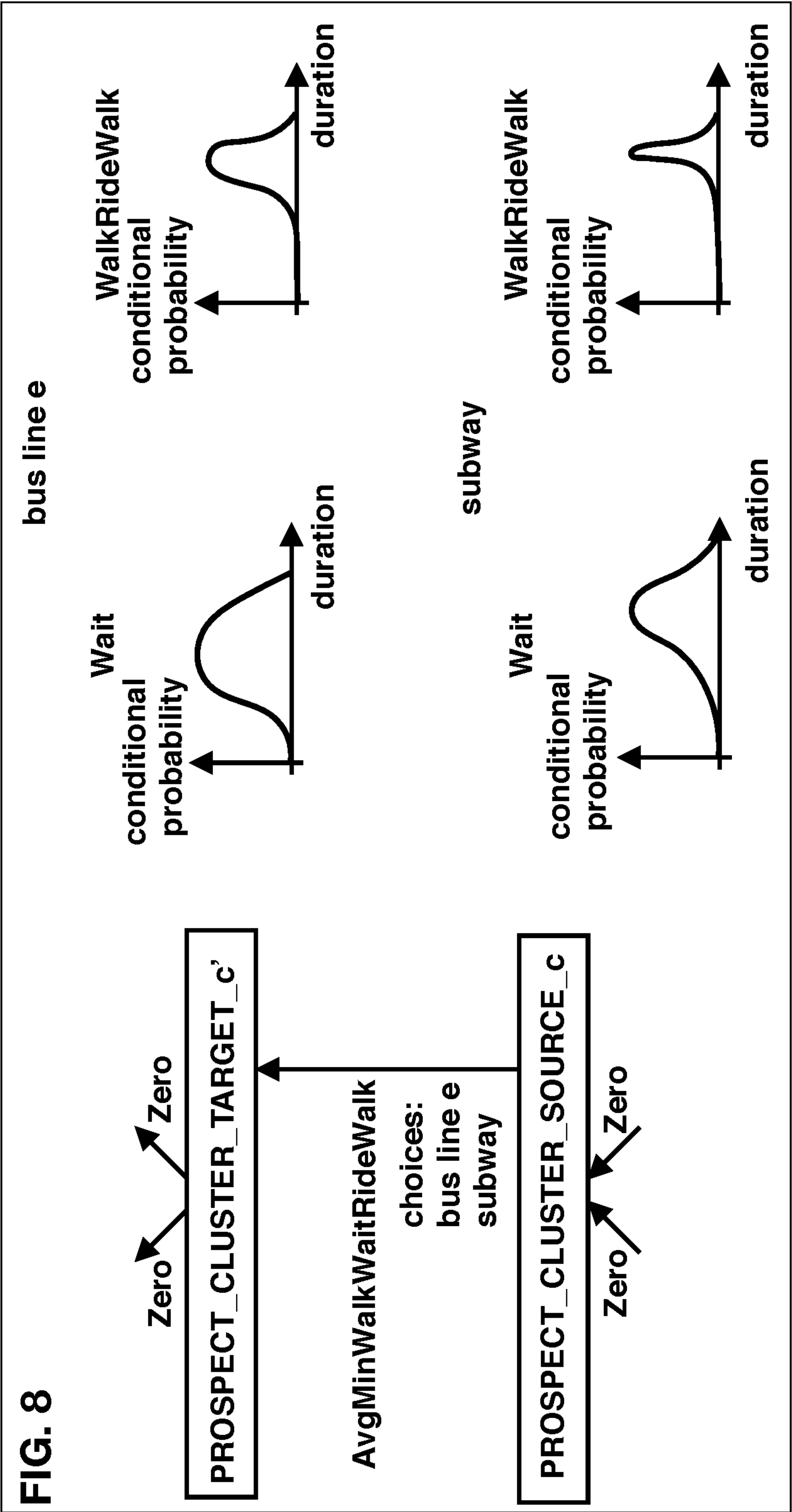


FIG. 9

```

let C and C' be two clusterings of bus stops and subway stations

for each cluster c' in C'
  add vertex PROSPECT_CLUSTER_TARGET_c'
  add edges
    PROSPECT_CLUSTER_TARGET_c' → BUS_STOP_b
    PROSPECT_CLUSTER_TARGET_c' → SUBWAY_STATION_s
  with zero weight, for every bus stop b and subway station s in cluster c'

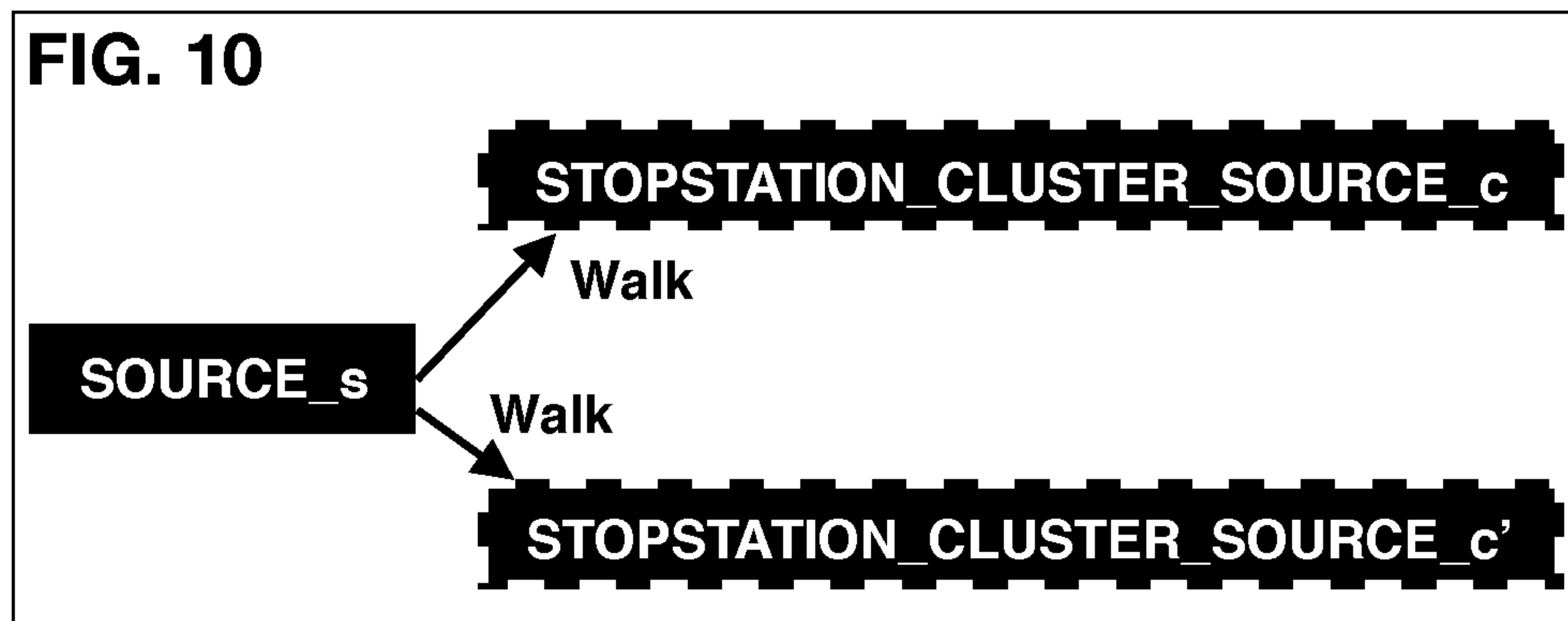
for each cluster c in C
  add vertex PROSPECT_CLUSTER_SOURCE_c
  add edges
    BUS_STOP_b → PROSPECT_CLUSTER_SOURCE_c
    SUBWAY_STATION_s → PROSPECT_CLUSTER_SOURCE_c
  with zero weight, for every bus stop b and subway station s in cluster c

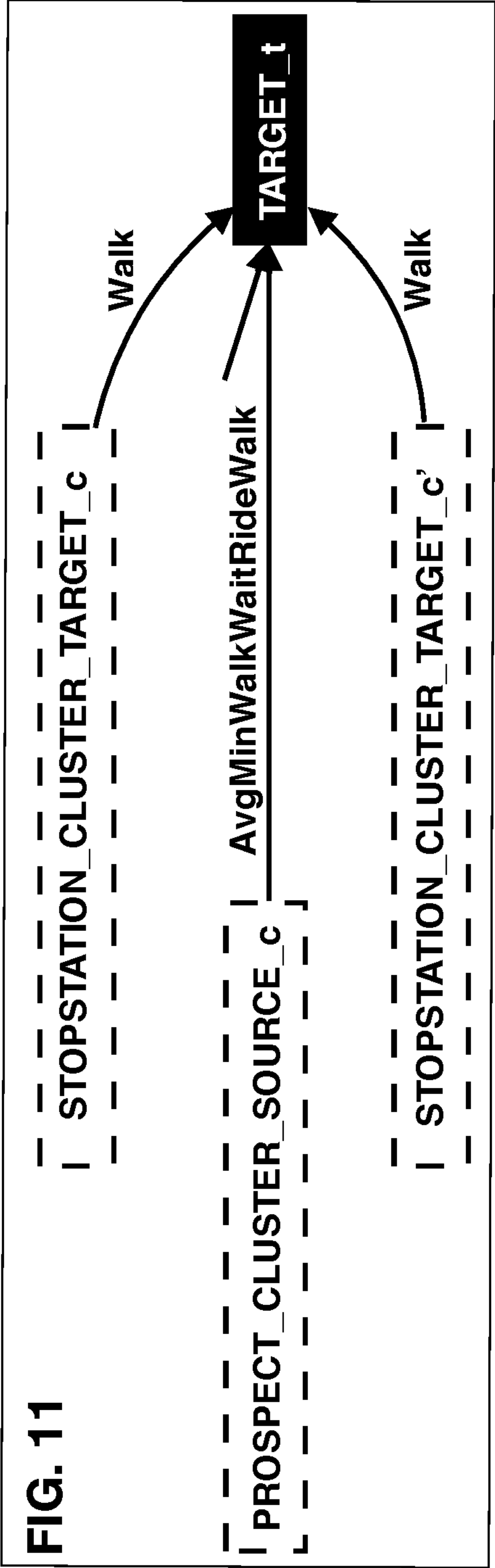
for each cluster c in C
  independentMap = empty map
  for each walk wHead from c to a nearby bus stop BUS_STOP_b
    perform a depth-first traversal of the graph starting at BUS_STOP_b
      to enumerate paths with edges WaitGetOn, RideSame, ..., RideSame, GetOff
    for each path
      let e be the bus line, and BUS_STOP_b' be the vertex at the end of the path
      let rideDuration = sum of weights of the RideSame along the path
      let maxWaitDuration = 2 * weight of the WaitGetOn on the path
      for each shortest walk wTail from BUS_STOP_b' to a cluster c'
        let moveDuration = duration of wHead + duration of wTail + rideDuration
        if not mapped at [c', e], or the mapped > moveDuration + maxWaitDuration / 2
          independentMap[c', e] = (maxWaitDuration, moveDuration)

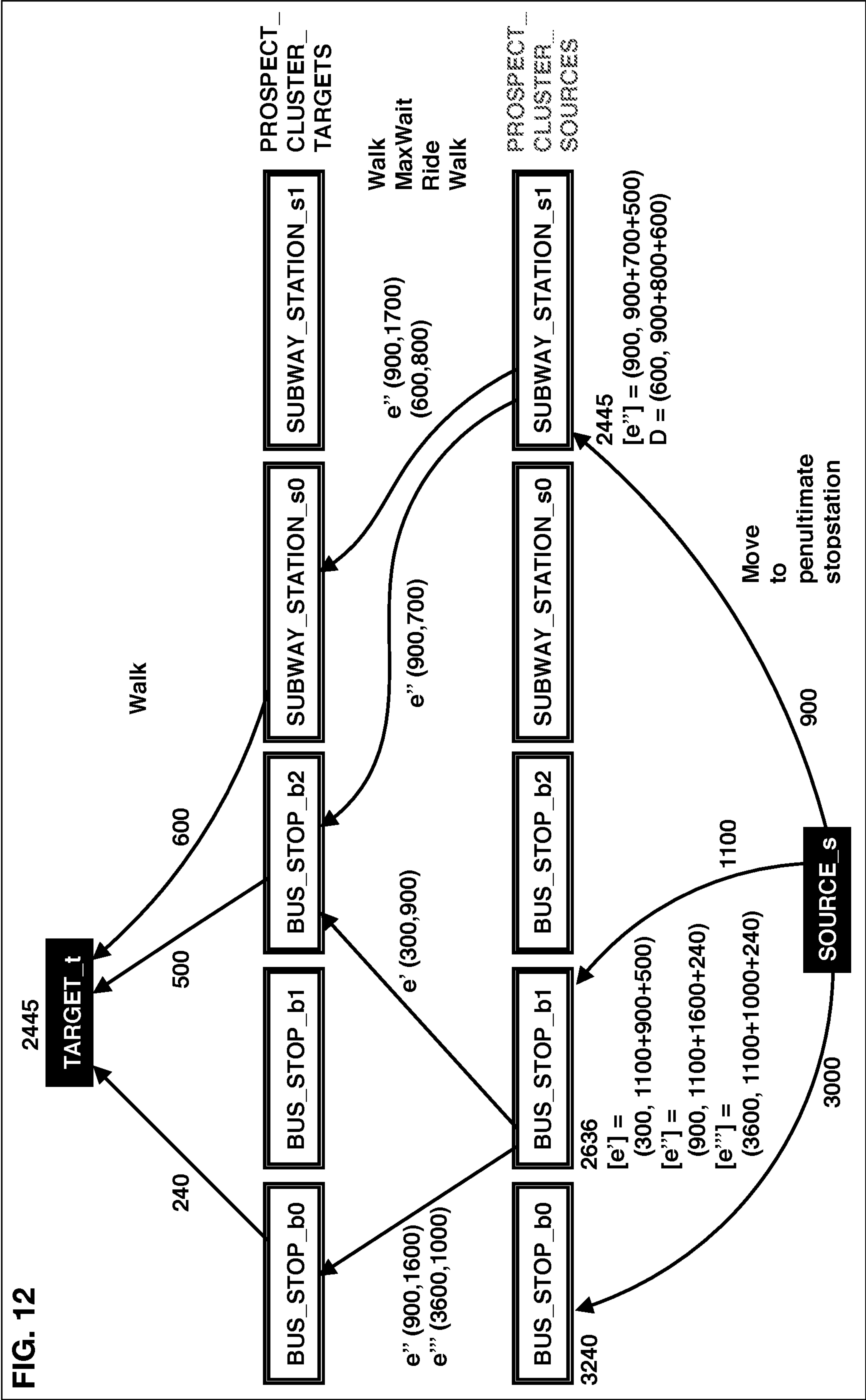
  dependentMap = empty map
  for each walk wHead from c to a nearby subway station SUBWAY_STATION_s
    perform a depth-first traversal of the graph starting at SUBWAY_STATION_s
      to enumerate paths with edges WaitGetOn, RideManyGetOff
    for each path
      let SUBWAY_STATION_s' be the vertex at the end of the path
      let rideDuration = weight of the RideManyGetOff on the path
      let maxWaitDuration = 2 * weight of the WaitGetOn on the path
      for each shortest walk wTail from SUBWAY_STATION_s' to a cluster c'
        let moveDuration = duration of wHead + duration of wTail + rideDuration
        append (maxWaitDuration, moveDuration) to dependentMap[c']

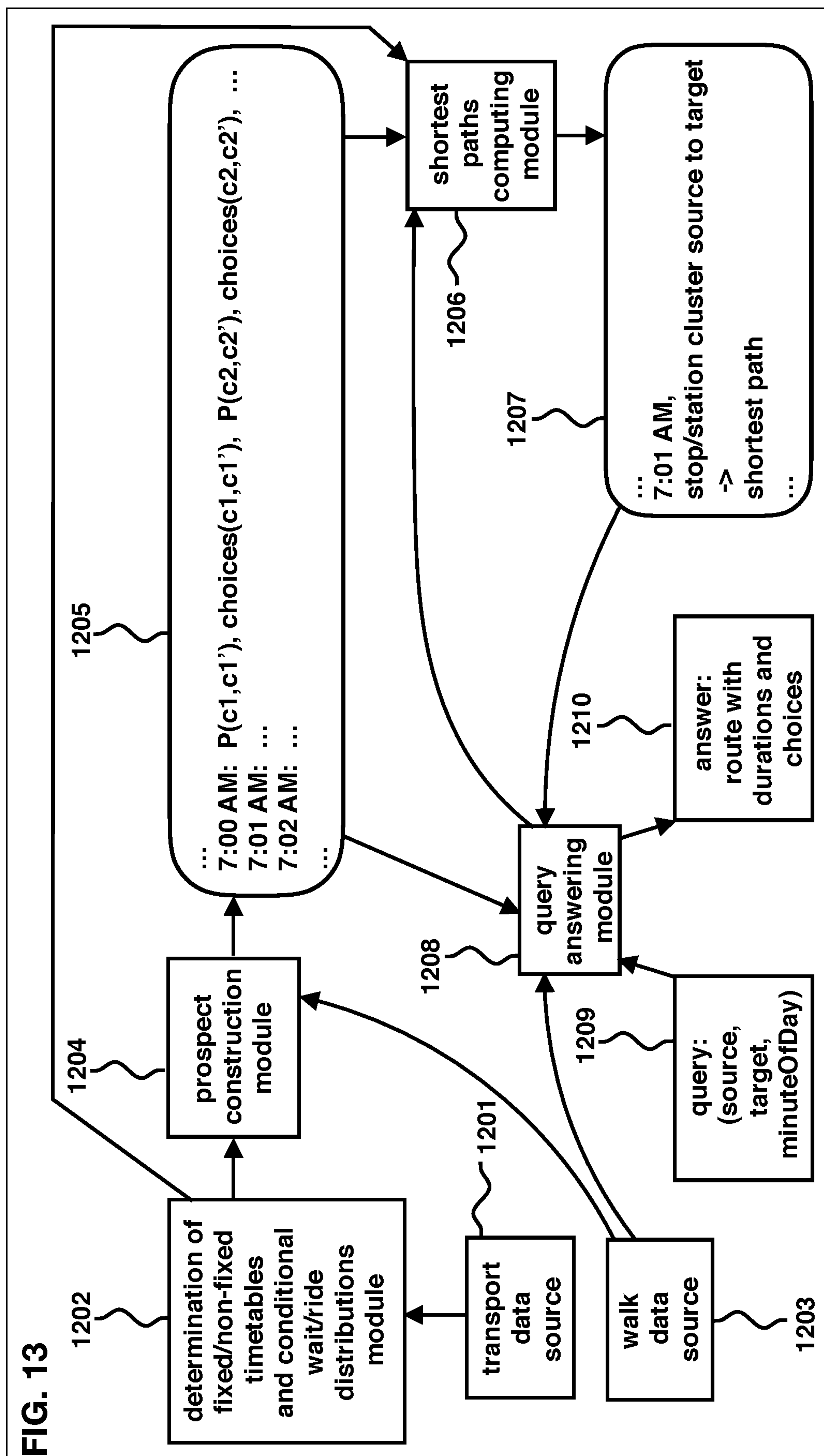
for each cluster c' in C'
  let there be exactly k bus lines e_1, ..., e_k mapped by independentMap with value
    independentMap[c', e_i] = (maxWaitDuration_i, moveDuration_i)
  we define k independent random variables T_1, ..., T_k so that
    T_i is uniform on interval [moveDuration_i, moveDuration_i + maxWaitDuration_i]
  prospectWeight = Expected[min{T_1, ..., T_k}]
  for each (maxWaitDuration, moveDuration) on the list dependentMap[c']
    let T be uniform on interval [moveDuration, moveDuration + maxWaitDuration]
    prospectWeight = min(prospectWeight, Expected[min{T, T_1, ..., T_k}])
  add edge PROSPECT_CLUSTER_SOURCE_c → PROSPECT_CLUSTER_TARGET_c'
  labeled AvgMinWalkWaitRideWalk with weight prospectWeight

```









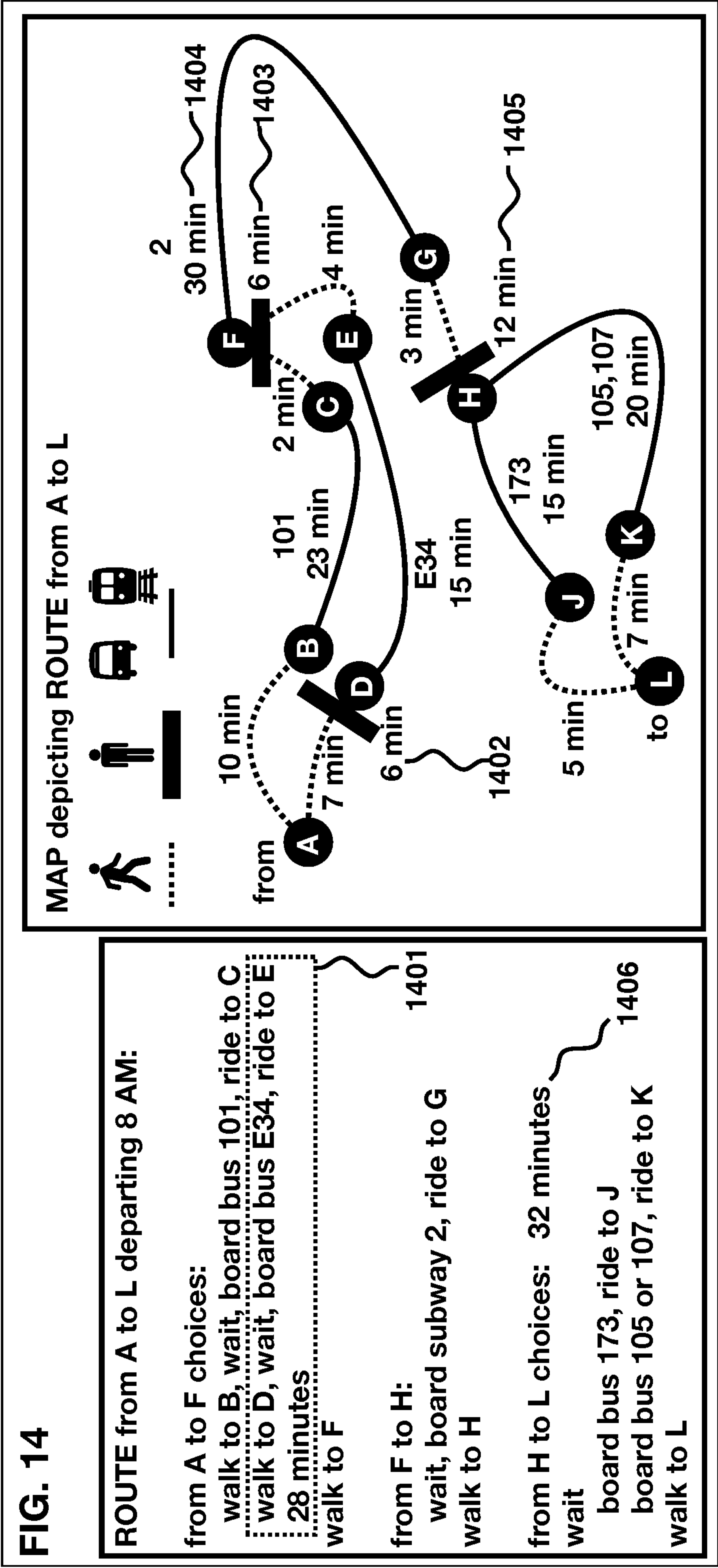


FIG. 5

