

Transportation System and Method for passenger and route scheduling

Field of the Disclosure

The present application relates to a transportation and routing application. Specifically, the application relates to a transportation management system and a transportation management method. More specifically, the present application relates to scheduling and routing of passengers in particular in a multi-stop scenario.

Background

Individual transport services in the recent are shifting from personally driving a vehicle to mobility as a service MAAS applications. Here, a person to be transported does not own a vehicle anymore but uses the provided mobility on demand, i.e. uses the service of being transported from point A to point B at a time of choosing. Such trips in turn may be of a private or semi-private nature. A private trip is one where the transport vehicle is exclusively scheduled for a passenger or a group of passengers belonging together, and is most of the time characterized that all passengers want to travel between the same points of origin and destination. A semi-private trip is characterized as one where a certain, mostly small number of people are pooled together and share a ride. Such a semi-private ride may be between the same points A and B for all passengers currently sharing the vehicle. Alternatively, a vehicle may schedule a trip with a plurality of points, allowing passengers to embark and disembark along the route of travel.

For providing such a transport service requires to plan and allocate a plurality of users to a plurality of vehicles that service the destinations the users desire. As such a solution to the routing and schedule allocation problem has to be provided. In particular in this context the so called vehicle routing problem (VRP) has to be considered.

The VRP may be described as a combinatorial optimisation problem that comprises finding the best routes and number of vehicles to visit a set of locations. Several variants of VRP are known, which differ from each other by their constraints. For example, one variant, known in the art as the "Travelling Salesman Problem", relates to only one vehicle but no constraints on the time and the order by which the locations are to be visited. Another variant, known in the art as the "Pick-up and Delivery Problem", places constraints on the order of location visits, as a pick-up location must precede its corresponding drop-off location. A further variant, known in the art as the "Capacitated Vehicle Routing Problem with Time Windows" (CVRPTW), considers that each vehicle has a capacity attribute and each trip request or booking has a capacity request and a preferred time window for pick-up and drop off. Most classic VRP variants also restrict the vehicles to start and end their routes at a depot.

A VRP problem is typically formulated as a graph, with nodes that represent the aforementioned depot, pick-up and drop-off locations, as well as connections that represent the route segments between successive nodes. The optimality of a solution is described or measured with a cost function. For example, the cost function may be the total distance of all routes, and the best solution may be the one that results in the minimum cost, or minimum total route distance. In some application scenarios, the cost may be the total driving time. In other scenarios, it may be designed to greatly favour low number of vehicles over total distance. The cost function may also relate to costs like fuel consumption, or an environmental impact of the use of a vehicle or a fleet, and may also include the actual cost for operating a certain vehicle per distance travelled and/or of passenger transported. The cost function may of course consider not just one but also a plurality of individual costs, which may be weighted appropriately.

Due to the enormous combinatorial search space of VRP problems, given computing resource and time constraint in reality, it is computationally prohibitive to find the true globally optimal solution. Therefore, VRPs are regularly solved by finding sub-optimal solutions in a realistic time constraint, using efficient combinatorial optimisation methods and meta-heuristics. A metaheuristic, or metaheuristic function or metaheuristic guiding function, which terms may be used interchangeably, is a higher-level procedure designed to guide, find, generate, or select a heuristic, i.e. a partial search algorithm, which may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity. Metaheuristics may not guarantee that a globally optimal solution can be found on some class of problems. Many metaheuristics implementations some form of stochastic optimization, so that the solution found is dependent on the set of random variables generated. In combinatorial optimization, by searching over a large set of feasible solutions, metaheuristics can often find good solutions with less computational effort than optimization algorithms, iterative methods, or simple heuristics.

For example, a metaheuristic method may guide a neighbourhood search method through multiple iterations to derive a solution. In each iteration, the neighbourhood search method finds a new solution in the search space around the current solution, and the metaheuristic method decides whether the new solution is acceptable. If acceptable and all constraints are met, the new solution becomes the current solution for the next iteration.

The most frequently adopted algorithm for solving combinatorial optimization problems is neighbourhood search.

This algorithm iteratively searches for locally or globally optimal solution in the solution space by creating permutations of the current solution according to specific rules to generate candidate solutions. A candidate solution is accepted if its cost is lower than the cost of the current best solution, and all constraints are satisfied. In general, the candidate generation strategies

depend greatly on the specifics and constraints of the problem. If properly designed, the strategies can greatly simplify constraint compliance checks during the permutation process.

Search space and solution space may in particular be understood as the set of all possible points (sets of values of the choice variables) of an optimization problem that satisfy the problem's constraints.

The search space is the initial set of candidate solutions to a particular optimization problem, before the set of candidates has been narrowed down. The narrowed down set of points satisfying the problem's constraints, in turn may be referred to as the solution space. A constraint is a condition of an optimization problem that the solution is required to satisfy.

One example of such a neighbourhood search is provided in Fig 11. The search starts with an initial solution that is generated and stored into a memory element like the solution storage element 1100. On every iteration, the current solution in the solution storage element 1100 is passed to different local search operator elements 1120. All enabled local search operators are executed in a loop. Operators use current solution to generate new neighbour solution. Every operator uses its own algorithm to mutate the current solution to produce one or more neighbour solutions. Typically, algorithms are simple and do not consider all constraints defined for the problem, so the generated candidate (neighbour) solution might or might not meet the constraints. All solutions are passed to filter element 1110 comprising one or a plurality of filter instructions, and filter element 1110 select only solutions that meet the constraints and have better objective function value, i.e. provide a better solution than the current solution. If such a solution is found, it is saved to solution storage element 1100 and the local search continues using new found solution as a reference for operator elements 1120. The operator elements 1120 are applied sequentially, and order by which the operators are applied is controlled by operator sequencer 1130. When filter element 1110 detects a new solution, the operator sequence can be started from the first operator. Local search stops if, after a full cycle of all operators, no new solutions were found.

Exemplarily, three types of operator elements 1120 in the local search step are described now, "2-Opt", "Cross" and "Exchange".

With "2-Opt", an operator tries to reverse all possible combinations of subsequences of every route to find a neighbour solution with a better objective value.

Figure 12 shows an exemplary embodiment of a neighbourhood search method with an operator element using "2-OPT" of the neighbourhood search.

Exemplarily, every route contains $N+1$ nodes, where n_0 is the starting node or location, e.g. a depot and n_N is the ending node or location, e.g. another (same or different) depot. A subsequence of this route $[n_F, n_L]$ contains all nodes from node F to and including node L , where F denotes the first node of the subsequence, and L denotes the last node of the subsequence. The target is to exhaustively generate all possible combinations of subsequences that satisfy three conditions:

Condition 1: The first node is before the last node. ($F < L$ – step 1210)

Condition 2: The first node is on the route, but excluding the depots. ($0 < F < N$ – step 1250)

Condition 3: The last node is on the route, but excluding the depots. ($0 < L < N$ – step 1230)

The algorithm begins at 1200 by initializing F and L of the current route to 1, which is the node after the starting depot of the current route. If condition 1 is satisfied (decision: yes), the method continues to step 1220 and creates a new candidate solution by reversing the subsequence $[n_F, n_L]$ of the route and then evaluates the associated objective value. Steps 1230 to 1260 generate new subsequences systematically while ensuring that the other conditions 2 and 3 are satisfied.

After all subsequences of the current route have been evaluated, step 1270 checks if the current route is the last route. If it is, the algorithm ends. Otherwise, the method returns to step 1200 and the steps are repeated for the next route.

With “CROSS”, an operator aims to exchange starting subsequences of a pair of routes to find a neighbour solution with a better objective value.

Figure 13 shows an exemplary embodiment of a neighbourhood search method with an operator element using “CROSS” of the neighbourhood search.

Exemplarily, for every pair of routes, the first route contains $N+1$ nodes, where n_0 is the starting location, e.g. a depot and n_N is the ending location, e.g. another (same or different) depot. A starting subsequence of this route $[n_1, n_L]$ contains all nodes from node 1 to and including node L , where L denotes the last node of the subsequence. Likewise, the second route contains $M+1$ nodes, where m_0 is the starting location, e.g. a depot and m_M is the ending location, e.g. another (same or different) depot. A starting subsequence of this route $[m_1, m_K]$ contains all nodes from node 1 to and including node K , where K denotes the last node of the subsequence. The target is to exhaustively generate starting subsequences for all possible pairs of routes, exchange them, and evaluate the resultant solution.

The method begins at step 1300 by picking a new pair of routes and initializing L and K to 1. Step 1310 creates a candidate solution by exchanging $[n_1, n_L]$ of the first route and $[m_1, m_K]$ of the second route, and evaluates its objective value. Step 1320 checks whether both L and K have reached the end of their respective route. If they have not (decision: no), L and K are concurrently incremented at step 1330 to produce a new pair of starting subsequences and continue from step 1330 for solution generation and evaluation. If they have (decision: yes), a new pair of routes is selected and the entire process is repeated. The process ends when all possible pairs of routes have been processed, which is determined in step 1340.

With “EXCHANGE”, an operator tries to exchange all possible pairs of nodes of every route to find a neighbour solution with a better objective value.

Figure 14 shows an exemplary embodiment of a neighbourhood search method with an operator element using “EXCHANGE” of the neighbourhood search.

Exemplarily, every route contains $N+1$ nodes, where n_0 is the starting location, e.g. a depot and n_N is the ending location, e.g. another (same or different) depot. A pair of nodes of this route is denoted by (n_F, n_S) , where F denotes the first node of the pair and S denotes the second node of the pair. The target is to exhaustively generate all possible pairs of nodes that satisfy three conditions.

Condition 1: The first node is before the second node. ($F < S$ – step 1410)

Condition 2: The first node is along the route, but excluding the depots. ($0 < F < N$ – step 1450)

Condition 3: The second node is along the route, but excluding the depots. ($0 < S < N$ – step 1430)

The method starts at step 1400 by initializing F and S of the current route to 1, which is the node after the starting location of the current route. If condition 1410 is satisfied, the method continues to step 1420 and creates a new candidate solution by exchanging n_F and n_S of the route and then evaluates the associated objective value. Step 1430 to 1460 generate new node pairs systematically while ensuring that the other condition 2 and condition 3 are satisfied.

After all pairs of nodes of the current route have been evaluated, step 1470 determines if the current route is the last route. If it is, the method ends. Otherwise, the method is repeated from step 1400 for the next route.

One classical meta-heuristic method is “Simulated Annealing”. “Simulated Annealing” is an iterative algorithm based on neighbourhood search. For every iteration of neighbourhood search a probabilistic mechanism that uses a parameter, e.g. a temperature parameter, to determine the probability of accepting a candidate solution is used. The parameter, exemplarily the temperature, starts at high initial value and is decreased over the course of neighbourhood search iterations according to a selected cooling scheme. As the parameter decreases, e.g. towards a temperature of “zero”, the probability of accepting candidate solution with higher energies (i.e. worse solutions) decreases. This property of accepting worse solutions in initial iterations enables “Simulated Annealing” to search for a globally optimal solution more extensively.

For routing problems, the solution cost is usually defined to be directly proportional to the energy, and an exponential cooling scheme is used. The probability of accepting higher-cost solutions is gradually lowered as the iterations progress. Better solutions with lower cost values are always accepted in every iteration.

Another neighbourhood search method is “Ruin and Recreate”. “Ruin and Recreate” ruins portions of the routes of a solution in a “Ruin” phase and then recreates them in a “Recreate” phase. One “Ruin” strategy is called “adjacency-based selection”, which picks a random node on the graph and removes nodes or sub-sequences of nodes in increasing distance from the node

until a defined number of nodes for this iteration are removed. The subsequent “Recreate” phase adapts the existing routes or create new routes to integrate the recently orphaned nodes.

Another “Ruin” strategy, which is especially effective for early stages of iteration, is “full-route selection”, which removes all nodes of one route, thereby decreasing the overall number of routes. Usually, a route with less constrained nodes is selected to be “ruined”. The “Recreate” phase then tries to insert the newly removed nodes into remaining routes.

Figure 7(a) to (c) show an exemplary embodiment of a “Ruin and Recreate” neighbourhood search method.

Figure 7(a) shows an intermediate solution in which three routes connect a set of nodes in an intertwined, sub-optimal fashion. By adopting an “adjacency-based selection” strategy, a circular area 7010 is defined around a random node 7000.

Figure 7(b) shows that a “Ruin” process has ruined all the connections that were fully contained in the circular area 7010, resulting in a few orphaned and/or disconnected nodes 7020.

Figure 7(c) shows that a “Recreate” process has taken place to generate a new solution, in which a more optimal set of routes has replaced the ruined routes.

While known solutions to the vehicle routing problem exist, there is yet no solution that optimally solved the problem in all circumstance. Thus, there is a need for an improved solution of the vehicle routing problem. The present disclosure has been devised in the light of the above considerations.

Summary of the Disclosure

At its most general, the present disclosure relates to a novel solution to solve the vehicle routing problem and its application in a transport system for ride scheduling.

According to a first aspect of the present disclosure, there is provided a computer-implemented method for route calculation in a multi-vehicle environment, comprising the steps of

- a) receiving booking data corresponding to a desired trip of a user, b) determining at least one first route data as current route data corresponding to the booking data, wherein each of the at least one first route data comprises at least one first route characteristic, c) determining at least one subsequent route data corresponding to the booking data and being based on the current route data, wherein each of the at least one subsequent route data comprises at least one subsequent route characteristic, and wherein a route characteristic is indicative of at least a quality of the respective route data, d) comparing the at least one subsequent route characteristic with the at least one route characteristic of the current route data, e) determining, based on the comparison, whether a quality of the at least one subsequent route data is increased in relation to the at least one current route data, f) selecting one of the current route data and the subsequent route data having the higher quality as the subsequent route data employed as route data solution or for the next iteration, and g) transmitting the route data solution to at least one of a vehicle as

a route solution and a user as a booking solution, wherein steps c) to f) are repeated until a defined exit condition is satisfied, wherein at least one of the at least one subsequent route data is determined by employing a metaheuristic function guiding a neighbourhood search function, wherein the neighbourhood search function is adapted to determine subsequent route data operating from the current route data, and wherein the search scope of the neighbourhood search function is set by a parameter of the metaheuristic function.

According to a second aspect of the present disclosure, there is provided a system for route calculation in a multi-vehicle environment, comprising a communication element adapted for receiving booking data corresponding to a desired trip of a user; a determining element adapted for determining at least one first route data as current route data corresponding to the booking data; wherein each of the at least one first route data comprises at least one first route characteristic; a determining element adapted for determining at least one subsequent route data corresponding to the booking data and being based on the current route data; wherein each of the at least one subsequent route data comprises at least one subsequent route characteristic; and wherein a route characteristic is indicative of at least a quality of the respective route data; a comparing element adapted for comparing the at least one subsequent route characteristic with the at least one route characteristic of the current route data; a determining element adapted for determining, based on the comparison, whether a quality of the at least one subsequent route data is increased in relation to the at least one current route data; a selecting element adapted for selecting one of the current route data and the subsequent route data having the higher quality as the subsequent route data employed as route data solution or for the next iteration; and wherein determining subsequent route data, comparing subsequent route characteristic, determining a quality increase and selecting route data with increased quality is repeated until a defined exit condition is satisfied; wherein at least one of the at least one subsequent route data is determined by employing a metaheuristic function guiding a neighbourhood search function; and wherein the neighbourhood search function is adapted to determine subsequent route data operating from the current route data; wherein the search scope of the neighbourhood search function is set by the metaheuristic function.

According to a third aspect of the present disclosure, there is provided a transport routing system for route scheduling in a multi-vehicle environment, the system comprising a communication element adapted for receiving booking data from at least one user, and adapted for receiving vehicle availability data of at least one vehicle; at least one processing element for determining at least one route data solution for the vehicle in accordance with the booking data of the at least one user; wherein the route data solution is determined by executing the method according to the present disclosure.

According to a fourth aspect of the present disclosure, there is provided a computer program product comprising instructions which, when the program is executed by a computer, cause the computer to carry out the method according to the present disclosure.

According to a fifth aspect of the present disclosure, there is provided a computer-readable
5 medium having stored thereon the computer program product of the present disclosure.

Generally, the present disclosure employs a novel implementation of the vehicle routing problem, in particular a combination of a particular metaheuristic function and a particular neighbourhood search method. The metaheuristic function may be employed globally on the search space to guide a search function to one or more regions of the search space that may
10 exhibit a preferred outcome in relation to other regions of the search space. A subsequent optimization may be performed with the neighbourhood search function. Such a neighbourhood search function may e.g. be embodied as a ruin and recreate search function.

In one embodiment of the invention, the metaheuristic function employs Simulated Annealing to iterate two neighbourhood search functions: In initial iterations, Ruin and Recreate
15 is employed as neighbourhood search function, while in late iterations, an alternative neighbourhood search function, such as a simple local search, e.g. hill climbing, is alternatively employed.

In another embodiment of the invention, the metaheuristic function employs Simulated Annealing to iterate a dynamically configured Ruin and Recreate method. E.g., the number of
20 nodes to ruin is configured to be directly proportional to a global parameter, e.g. the temperature parameter of Simulated Annealing. In other words, the number of nodes to be ruined or the ruin radius or ruin area is large initially but is gradually reduced over the course of iterations.

One of the key properties of a ruin and recreate neighbourhood search function is the ruin area. In known applications, the ruin area is statically configured, while the ruin area of the present
25 disclosure is dynamically allocated or changed, in particular dependent on a parameter, like the temperature parameter of a metaheuristic function. A large ruin area is advantageous for initial iterations, which are designed to facilitate a global search for optimal solution. However, as the solution becomes better and approaches a plateau and becoming more or less optimal over the course of the iterations, a large ruin area becomes ineffective in producing small refinement to
30 the solution. On the other hand, configuring a small ruin area at the outset can cause the solution to get stuck at a local minimum during very early stage of iterations. To start a ruin and recreate search function, an initial or seed node may be chosen by random. Further, it is conceivable that multiple ruin and recreate search functions centring on different seed node are performed in parallel and the best result is accepted.

35 It should be noted that the aforementioned Simulated Annealing metaheuristic function and Ruin and Recreate neighbourhood search function only serve to illustrate possible embodiments of the invention. It's understood that other metaheuristic functions, like the "Tabu"

metaheuristic function and neighbourhood search functions exhibiting similar characteristics also fall within the scope of the invention. Similar characteristics may e.g. be neighbourhood search methods with known, or which may be configured to achieve “jumps” of certain magnitude in the solution space. E.g. metaheuristic functions characterised by “big jumps” may be coupled with
 5 neighbourhood search functions characterised by “small jumps” in a complementary manner. Configurable methods can be dynamically configured so that they exhibit different jump characteristics at different stages of iterations.

The present invention also provides a transportation system for ride scheduling and pooling of users. In a scenario of the present disclosure with a plurality of vehicles and a dynamic
 10 passenger-trip allocation, the individual route of a vehicle is changing depending on the passenger-trip allocation, i.e. the allocation of certain passengers to a vehicle in accordance with their desired pick-up and drop-off locations. To optimise a certain mobility as a service offering, a preferred allocation of passengers to a plurality of a vehicles for a given point in time has to be provided. Such a preferred allocation may result in a minimum overall travel time of a certain
 15 passenger or of an average minimum overall travel time of all certain passengers. Likewise, a reduced average travel time goes hand in hand with a reduction in fuel consumption, in particular a reduction of a fleet consumption of all relevant vehicles (e.g. those currently in operation) participating in the mobility service, and thereby reducing emissions of the vehicles. Likewise a total amount of distance travelled can be reduced, thereby prolonging the operational lifetime of
 20 vehicles and reducing maintenance costs.

The system and method of the present invention relates to a transport service, in particular for pooling of passengers and scheduling of trips in a multi-vehicle scenario. Users can inquire for a trip ahead of time, e.g. register for a trip the next day at 8am that takes them from point A to point B. The invention considers all such trip inquiries or prior bookings (arriving ahead of time)
 25 and schedules, depending on at least some of a requested departure or pick-up location, a requested destination or drop-off location, a requested pick-up time or time window, a requested drop-off time or time window, a capacity and the like, a user and car allocation, to most efficiently service all or at least the majority of requests from the users of the transport service.

The system and method schedules the trips and informs a user whether his inquiry has
 30 been accepted and/or provides trip details, which may deviate to some degree from the initial request. E.g. a pick-up location and pick-up time may have been changed slightly to better accommodate surrounding trips. This may mean that the user is offered not to be picked up at his preferred location at e.g. 8:05 am, but at a location 300m away at 8:15am, while still maintaining a requested drop-off time (window) at the requested drop-off location (area). Such a slight change
 35 in time and location may allow to further optimize a pooling service over a mere trip sharing application, as it allows to pool more users than would be possible if the service would more stringent keep to the requested trip details.

The system may generate a plurality of trips that are presented to a user as options, from which the user may choose a preferred trip. All presented trips may correspond to the requested trip to some extent and may deviate from the requested trip differently. E.g. one option may present the user with a pick-up location matching the requested pick-up location but some time earlier than requested, while a second option may offer a trip within the requested time window but with a pick-up location that is further away. The user may then decide to accept one the provided options or reject all options. It is conceivable that depending on the plurality of accepted trips by user of the system ahead of time, that a further optimization is achievable. E.g. a vehicle allocation may be changed, i.e. vehicles may be chosen for a certain trip dependent on the now required capacity.

The scheduling by the transportation system may operate in a pre-booking phase, where a plurality of bookings are collected and pre-processed. The data is fed to the system to perform a first VRP calculation, to produce an initial set of routes to service the bookings. When in operation, i.e. the bookings are currently serviced, the system may operate in an on-demand phase. In this phase, the system continuously receives additional bookings and inserts them into the existing routes, by repeating the VRP calculation. A plurality of events are conceivable that trigger an update and thus may potentially alter calculated routes. Such events may be a new booking, a booking cancellation, a new group of bookings accumulated over a time interval, a scheduled calculation event, a road closure, a road accident and or a breakdown of a vehicle currently in operation and servicing users.

The present invention may however not only be used in such a pre-scheduling environment but may also be applied for scheduling presently incoming trips for allocation to a vehicle fleet or also only a single vehicle currently in operation. The present invention may enable vehicle passengers of e.g. a bus service to request pick-ups and drop-offs at any bus stop in certain areas of a city, for example by using a dedicated mobile app. The time may be unrestricted, or may be limited e.g. at off-peak times, for example between 11am to 3pm, and from 8pm onwards, when demand is lower and a fixed schedule public transportation service may no longer be profitable. As off-peak buses may carry fewer passengers, the goal may be seen to reduce costs by replacing a public transportation service with an on-demand service. Regular buses may still service the selected routes but at lower frequencies. The present invention may allow to order an individual ride immediately or pre-book a ride for a defined time, e.g. up to two weeks, in advance. Fares may be calculated on a distance-based pricing model and thus reflect the actual travel.

A preferred embodiment of the present invention may be seen as follows, which may apply to all technical details given above and below. The method and the system of the present disclosure may use the metaheuristic function to guide a search function to iterate a candidate solution with adaptive Ruin and Recreate (i.e. reducing adjacency range with a parameter, e.g.

the temperature of simulated annealing as the metaheuristic function). When the parameter reaches a threshold T_s , the search function switches from adaptive Ruin and Recreate to e.g. local neighbourhood search, possibly using at least one of 2-opt, Cross and Exchange operators as described herein.

5 A characteristic of route data, or route characteristic, may be a quality of the route data. The characteristic or quality indicator may be understood as any indicator or information element able or used to express that one route data comprise route features, e.g. nodes of a route, which are better suited to service a particular demand than a route data having a lower quality. A route with higher quality may e.g. be a shorter or quicker route, may consume less fuel or require a
10 lower total number of vehicles to provide certain service. A route quality indicator may also be a number of people serviced by a single/the total number of vehicles, with a higher quality being expressed by an increased number of passengers in a vehicle/all vehicles, which allows to better utilise a given capacity, thereby increasing use.

Booking data may in particular be understood as any indicator or information element
15 related to a booking, in particular to allow determination of a route data. Booking data may be directly related to the environment, e.g. a real life scenario like the city surroundings where the service shall be provided, or may be metadata used for the calculation of route data in the context of the present method and system. As such, metadata may be transformed data obtained from the actual booking data, but e.g. taken out of the environmental context.

20 An exit condition may be a condition where the quality of a route data is maximized, or at least exceeding or meeting a defined threshold value, i.e. the route is of defined or required quality. Alternatively, the condition may be a threshold value related to a number of iterations of repetitions or a time used for the calculation, which should not be exceeded, or after which the repetition of iterations is halted. Likewise, an exit condition may be a condition considering the
25 guiding parameter of the metaheuristic function, e.g. temperature with simulated annealing or tabu list size for TABU search, and a related threshold value.

According to a further embodiment of the present disclosure, the metaheuristic function may comprise one of a "simulated annealing" search function or a "Tabu" search function.

Using one of said metaheuristics may provide one or a plurality of preferred local solution
30 for a global search of the solution space.

According to a further embodiment of the present disclosure, the neighbourhood search function may comprise at least one of a "ruin and recreate" search function, an adaptive "ruin and recreate" search function and a "local neighbourhood search" search function.

Using one of said neighbourhood search functions allows to find a preferred solution in
35 the solution space, while being guided by the metaheuristic function. In particular, the search function may start out with the metaheuristic function guiding an adaptive "ruin and recreate" search function

According to a further embodiment of the present disclosure, the metaheuristic function may comprise one of the “Simulated Annealing” search function and the “Tabu” search function, and the neighbourhood search function may comprise one of i) the “ruin and recreate” search function and the “local neighbourhood search” search function; and ii) the adaptive “ruin and
5 recreate” search function.

Using a described combination provides a preferred solution to the optimization problem in the solution space over executing the respective search functions independently. Further, using the described combinations provide better solutions to an optimization problem within the content of this disclosure in the solution space that known implementations.

10 According to a further embodiment of the present disclosure, in i) or iii) the respective “ruin and recreate” search function may be employed for a first number of repetitions and the “local neighbourhood search” search function may be employed for a second number of repetitions.

In other words, the present method may switch the neighbourhood search function while iterating. A first neighbourhood search function may be used, e.g. “ruin and recreate” or adaptive
15 “ruin and recreate”, until a certain condition is met. In the context of simulated annealing, the temperature parameter may reach a defined first temperature value, while in the context of Tabu, the parameter may be a Tabu list size. After reaching the parameter value, the method may subsequently continue the neighbourhood search with a different, e.g. the second neighbourhood search function, exemplarily “local neighbourhood search”. Alternatively, the order of the
20 neighbourhood search functions may be reversed.

According to a further embodiment of the present disclosure, setting the neighbourhood search scope by the metaheuristic function may comprise reducing the adjacency range of the adaptive “ruin and recreate” search function. In other words, the ruin area may depend on the parameter value of the metaheuristic function.

25 According to a further embodiment of the present disclosure, the metaheuristic function may comprise a parameter to be reduced within the progression of the repetition of at least steps c) to e), in particular a temperature parameter or a tabu list size; and wherein the adjacency range may be reduced dependent on the parameter.

In other words, the global parameter, e.g. the temperature parameter or the tabu list size
30 of the metaheuristic function sets the adjacency range of the ruin area. With progression of the optimization, the parameter is gradually reduced, e.g. the temperature is reduced, and consequently, the ruin area is reduced in size. This provides the functionality that with progressing optimization the search radius or adjacency range of the neighbourhood search function is reduced, in particular the distance between a previous/current solution and a new possible
35 solution to be evaluated is reduced, to avoid large jumps in the solution space when nearing the end of the metaheuristic function.

According to a further embodiment of the present disclosure the communication element further may be adapted for transmitting the route data solution to at least one of a vehicle as a route solution and a user as a booking solution.

According to a further embodiment of the present disclosure, the communication element
5 may be further adapted to provide the route data solution as trip data of the determined route data to the user; and/or to provide the route data solution to the at least one vehicle.

In other words, one or more found solutions may be provided to the user, e.g. as an offering from which the user can choose. The user can thus review the different trips or routes of the trips, and may choose one that best matches a particular preference of the user. E.g. two trips
10 may be provided to the user, which differ slightly regarding pick-up location and pick-up time, while meeting the overall constraints of the user like departure and arrival time window and time location. Likewise, once a trip has been accepted by the user, the allocated vehicle or the driver of the allocated vehicle receives information about the trip. The information may be individualized or may be combined information, like "pick-up three passengers at location A at time B".

According to a further embodiment of the present disclosure, the route data solution may
15 comprise at least one of a requested pick-up location, an allocated pick-up location, a requested drop-off location, an allocated drop-off location, a pick-up time, a pick-up time window, a drop-off time, a drop-off time window, a required capacity, a servicing sequence, an assigned vehicle and an assigned driver and/or wherein the vehicle availability data may comprise at least one of a
20 driver schedule, an average vehicle speed, a vehicle capacity and a vehicle location, wherein the vehicle capacity in particular may comprise a number of human passengers, a number of animal passengers, a size requirement of a good to be transported, a wheelchair requirement, a bed requirement, a container size and size of a required holding area.

According to a further embodiment of the present disclosure, the communication element
25 may be further adapted to receive road data and to employ the road data for determining the at least one route, in particular wherein road data may comprise at least one of road network information, traffic service information, road closure information and traffic information.

In other words, the transportation system, when scheduling routes/trips, may receive information regarding a current road condition or situation. E.g. a closed road due to road
30 maintenance may require a significant detour resulting in an increase in travel time. Likewise, information regarding traffic congestion may be received by the transport system, reflecting a current road situation, e.g. due to an accident. The transport system may thus utilize said information when determining the routes/trips. Such information may also be employed when servicing a current schedule or user, by e.g. allocating a different vehicle that need not travel via
35 the impacted traffic area in order to service a user, thereby avoiding the delay.

According to a further embodiment of the present disclosure, the transport system may further comprise a booking system, wherein the booking system may comprise a booking

application, wherein the booking application may be adapted to receive at least one booking data from a user via a mobile computing platform, a smartphone, a smart (digital) watch, a tablet, a laptop, a short messaging service (SMS), a mobile messaging application, a centralized booking portal, a telephone, and a voice call system.

5 In other words, the transportation system may employ a booking frontend for communication with the user of the system to provide a defined interface for registering trip requests. The frontend may e.g. provide a user interface for display on a device screen, where the user enters the trip preferences in a standardized and in particular machine readable manner, to facilitate reception of trip requests.

10 According to a further embodiment of the present disclosure, the transport system may be adapted for at least one of driver operated vehicles, autonomous vehicles, terrestrial vehicles, aerial vehicles, aquatic vehicles, amphibian vehicles and triphibian vehicles.

According to a further embodiment of the present disclosure, determining route data may further comprise pre-processing, by a pre-processing element, pick-up locations and drop-off
15 locations into permitted pick-up locations and permitted drop-off locations, in particular wherein at least some of the permitted pick-up locations and permitted drop-off locations may be associated with a weighting function; wherein the locations closer to the requested locations and/or statistically popular locations may be assigned a higher weight parameter than locations further away from the requested locations, and wherein the weighing function may be employed
20 when determining permitted pick-up and permitted drop-off locations.

Such pre-processing may reduce an initial search space or solution space in order to start only with allowed variables in the solution space. E.g. a desired pick-up time of 08:48 may be allocated to a pick-up time of 08:50 or to a time window of 08:50 to 08:55, or 08:45 to 09:00. Likewise, an arbitrary requested pick-up location may be allocated to a number of pre-set pick-up
25 locations in the vicinity of the requested pick-up location, to reduce complexity and to avoid a plurality of short trips around a certain pick-up area. Any requested user parameter, e.g. pick-up time and pick-up location, may be weighed depending on to what extent the parameter coincides with an allocated parameter. E.g. a potential pick-up location further away from a requested pick-up location may receive less weight than a potential pick-up location closer to a requested pick-up
30 location.

According to a further embodiment of the present disclosure, determining route data may further comprise changing the allocated pick-up and/or drop-off locations of at least one route data.

In other words, the transportation system may adapt a generated route of a potentially
35 already allocated trip, e.g. a trip accepted by a user. The system may change the route based on certain parameters, e.g. external input, like current traffic data or new trip requests, which can be served better by a different solution in the solution space. The at least one changed route may be

communication to the user, which may accept or reject the change, possibly rejecting the trip at all. Depending on the service contract, the user may be required to accept such changes in trip allocation. If a new booking is inserted into an existing route, the route regularly changes or updates, but additional constraints may be imposed during processing to keep certain parameters of the existing offers to currently scheduled users, e.g. the pick-up and drop-off locations, unchanged. The pick-up and drop-off times may change, but they may still be required to be within the time windows promised to the offered users.

Two scenarios for updating or changing allocated locations are conceivable. In a first pre-booking scenario, a plurality of received bookings are processed quasi simultaneously. Offers, i.e. trip data determined to be acceptable or serviceable, are sent to the users after all the routes have been finalised. In case a new solution to the routing problem required a change in location allocation, in the pre-booking scenario, such occurs before the offers are sent to the users, and as such a change in location allocation is transparent to the users.

In an on-demand scenario, in which a new booking may be inserted into existing offers (i.e. offered/accepted bookings), the pre-processing, routing engine and post-processing impose additional constraints that have to be met by the routing method, e.g. allocated locations of offers, may be required not be changed, and updated pick-up and drop-off times of existing offers, as a result of (additionally) serving the new booking, may be required to still be within the promised time windows.

According to a further embodiment of the present disclosure, changing the allocated pick-up and/or drop-off locations may further comprise the system being adapted to identify a plurality of permitted locations around each allocated location, identify all permitted transitions between said permitted locations, determine a shortest route with said permitted transitions, and change the allocated pick-up and/or drop-off locations to the permitted locations of the permitted transitions of the derived shortest route.

In other words, the transport system may determine a weighing factor by itself when considering the change of allocated pick-up and/or drop-off locations. Here, exemplarily, the shortest route of said permitted transition is employed as weighing factor, but further considerations may be taken into account, generally determine the weighing factor for said permitted transition, like e.g. time required for the transition.

According to a further embodiment of the present disclosure, the transport system may further be adapted to collect commute preferences comprising preferred pick-up location, preferred drop-off location, preferred pick-up time and preferred drop-off time, determine a transit pattern from said commute preferences, at least comprising at least one popular location, and change an allocated pick-up or drop-off location to favour popular locations.

In other words, a user may provide a set of commute preferences, which may be stored by the system in a suitable database, and which preferences are used when determining a route

for a user. The system may in particular favour popular locations, i.e. locations which are highly frequented by a plurality of user or which are easily accessible by the users, and which thus in particular beneficially impact the operation of the transport service.

Summary of the Figures

So that the invention may be understood, and so that further aspects and features thereof may be appreciated, embodiments illustrating the principles of the invention will now be discussed in further detail with reference to the accompanying figures, in which:

Figure 1 shows an exemplary embodiment of a method combining a metaheuristic function and a neighbourhood search function;

Figure 2 shows an exemplary embodiment of a method employing simulated annealing as metaheuristic function and two neighbourhood search functions according to the present disclosure;

Figure 3 shows an exemplary embodiment of a method employing simulated annealing as metaheuristic function and an adaptive neighbourhood search function according to the present disclosure;

Figure 4 shows an exemplary embodiment of the transportation system according to the present disclosure;

Figure 5 shows a variant of the exemplary embodiment of the transportation system of Figure 4;

Figure 6 shows an exemplary embodiment for finding the minimum length route in the transport systems according to the present disclosure;

Figures 7a to 7c show an exemplary embodiment of a “Ruin and Recreate” neighbourhood search method;

Figure 8 shows an exemplary embodiment of a method employing “Tabu” search as metaheuristic function and a neighbourhood search function according to the present disclosure;

Figure 9 shows an exemplary embodiment of a method employing a dynamic “Tabu” search as metaheuristic function and a neighbourhood search function according to the present disclosure;

Figure 10 shows an exemplary embodiment of a method employing a metaheuristic function guiding an adaptive “Ruin and Recreate” neighbourhood search function and a local neighbourhood search function according to the present disclosure;

Figure 11 shows an exemplary embodiment of a neighbourhood search method;

Figure 12 shows an exemplary embodiment of a neighbourhood search method with an operator element using “2-OPT” of the neighbourhood search;

Figure 13 shows an exemplary embodiment of a neighbourhood search method with an operator element using “CROSS” of the neighbourhood search;

Figure 14 shows an exemplary embodiment of a neighbourhood search method with an operator element using “EXCHANGE” of the neighbourhood search;

Figure 15 shows an exemplary embodiment of a ruin part of a “Ruin and Recreate” neighbourhood search method according to the present disclosure; and

5 Figure 16 shows an exemplary embodiment of a recreate part of a “Ruin and Recreate” neighbourhood search method according to the present disclosure.

Detailed Description of the Figures

Figure 1 shows an exemplary embodiment of a method combining a metaheuristic function and a neighbourhood search function. In particular, Figure 1 shows a combination of “Simulated
10 Annealing” and “Ruin and Recreate”.

As described, a “solution” of a vehicle routing problem comprises a set of routes, one route for each vehicle. Each route further comprises a sequence of nodes ordered according to the time by which they are serviced by the vehicle.

Step 100 sets the initial temperature for “Simulated Annealing”. Further, step 100 also
15 creates a simple initial solution for the iterative neighbourhood search function and records the initial solution as the best solution. One possible initial solution may be, e.g., ordering all nodes chronologically by their pick-up time and assigning them to a fleet of vehicles with a “greedy insertion” approach. The initial solution should comply with the problem constraints. Subsequently, step 110 executes a “Ruin and Recreate” method by ruining parts of the last
20 accepted solution and then recreating a new candidate solution. Step 120 checks the candidate solution, either the initial solution or the solution created by the Ruin and Recreate algorithm, for acceptability. If some constraints are not met, e.g. doesn’t meet the required pickup times, to fulfil the planned schedule, this candidate solution is rejected and a new candidate solution is generated based on last accepted solution. If the initial solution fails the constraints check, then
25 the algorithm may either try some other initial solution algorithm or fail.

If the candidate solution satisfies all the constraints, step 130 subjects it to a further test by computing the acceptance probability $P(T, S, S_{best})$ of the candidate solution with a random, uniformly distributed value in range [0, 1]. The acceptance probability is an exponential function of the energy difference between the candidate solution energy(S) and the best solution
30 energy (S_{best}), divided by the current temperature T .

$$P(T, S, S_{best}) = \exp\left(\frac{\text{energy}(S) - \text{energy}(S_{best})}{T}\right)$$

If the random value is lower than the acceptance probability, the candidate solution is accepted at step 140, otherwise the candidate solution is rejected and the algorithm continues with step 170. It should be noted that, probabilistically, step 130 allows a solution worse than the
35 best solution to be accepted as the current solution, as part of the mechanism of Simulated

Annealing to perform a global search during early iterations. As temperature lowers in subsequent iterations, it gets harder to accept worse solutions.

For the accepted solution, step 150 calculates the cost of the candidate solution, $\text{Cost}(S)$, and compares it with the cost of the best solution, $\text{cost}(S_{\text{best}})$. If the candidate solution cost is lower, then the candidate solution replaces the best solution in step 160.

To enhance readability, the solution processing steps from 120 to 160 will henceforth be encapsulated in a 'Temperature dependent Solution Selection' step 115 in subsequent illustrative figures.

Step 170 decreases the temperature of the system according to a chosen cooling scheme. Step 180 checks the new temperature of the system. If it drops below the pre-configured minimum value $T_{\min}$, the algorithm stops and presents the last accepted solution as the resulting/final solution in step 185. Otherwise, the method repeats the Ruin and Recreate step in step 110.

Figure 2 shows an exemplary embodiment of a method employing simulated annealing as metaheuristic function to guide two neighbourhood search functions according to the present disclosure. In particular, an embodiment of the invention is shown that uses Simulated Annealing as the metaheuristic function that guides two neighbourhood search methods operating in tandem: a Ruin and Recreate method and a Local Search method.

The functionalities of steps 200, 210, 215, 220, 230, 235 mirror steps 100, 110, 115, 170, 180 and 185 of Figure 1 respectively. Like step 130, step 230 checks whether the temperature T is below a threshold $T_{\min}$. If the condition is true, the method terminates and returns the last solution as the final solution in step 235. Otherwise, the process compares the temperature with another threshold T_s at step 240. If the temperature is above T_s and further Ruin and Recreate iterations are possible, the process performs the next "Ruin and Recreate" iteration at step 210, which potentially introduces large jumps in the solution space. Otherwise, if the temperature is below T_s , the process switches to a "Local Search" search function at step 250, which introduces more refined, smaller jumps in the solution space. Such a "Local Search" search function may e.g. be hill climbing or steepest descent. Step 240 may thus be seen as the step where the condition to switch from the first search function, here exemplarily ruin and recreate (or adaptive ruin and recreate), to a second search function, here exemplarily "Local Search" is met. The solutions produced by "Local Search" is also subjected to temperature-dependent processing at step 260, just like the solutions by "Ruin and Recreate" at step 215.

Figure 3 shows an exemplary embodiment of a method employing simulated annealing as metaheuristic function and an adaptive neighbourhood search function according to the present disclosure. In particular, an embodiment of the invention is shown that uses Simulated Annealing as the metaheuristic that guides an adaptive Ruin and Recreate method. An adaptive Ruin and Recreate search function may in particular be a Ruin and Recreate search function based on adjacency range reduction. In other words, the adjacency range of the adaptive Ruin

and Recreate search function may reduce over time dependent on the respective parameter of the metaheuristic function, e.g. the temperature of simulated annealing or the tabu list size of Tabu search.

The functionalities of steps 300, 315, 320, 330, 335 mirror steps 100, 115, 170, 180 and 185 of Figure 1 respectively. Furthermore, the Ruin and Recreate search function at step 310 is adaptive to a global parameter of the metaheuristic function, e.g. the temperature parameter. For example, if “Ruin and Recreate” is the neighbourhood search function and an “adjacency-based selection” strategy, as illustrated in Figure 7a to c, is employed, the adaptive “Ruin and Recreate” neighbourhood search function reduces the number of nodes to ruin 7010 in Figure 7a according to the temperature of parameter, to gradually shrink the area of ruin and thereby, the jump magnitude between the old and new solution in the solution space.

Figure 15 shows an exemplary embodiment of a ruin part of a “Ruin and Recreate” neighbourhood search method according to the present disclosure.

The Ruin phase uses an adjacency-based string ruin algorithm, whereas the Recreate phase adopts a greedy insertion algorithm. Here, a “string” refers to a consecutive set of nodes along a route.

For each iteration of the overarching method, step 1500 calculates the target number of strings to ruin, denoted by N , based on the average length of the routes in the current solution. In general, a long average route length means fewer vehicles are required, but longer strings should be ruined.

Step 1510 randomly selects a starting node, also known as the seed node, and a route to which the seed node belongs. Step 1520 calculates the number of consecutive nodes along the route to ruin, also known as the ruin string length, for the selected route. The ruin string length is randomly chosen from a predetermined range.

The ruin string is then processed by one of two possible ruin algorithms depicted by steps 1530 and 1540. The choice is randomly decided at every iteration. Step 1530 performs a string ruin, which ruins the entire string of nodes. Step 1540 performs a split-string ruin. This algorithm randomly selects two separate sub-strings along the selected route. The total combined length of the sub-strings is the ruin string length decided at step 1520. One of the sub-strings contains the selected seed node. Typically, the two sub-strings are selected such that the interval between them is large, and one of the sub-strings is near the end of the route.

Step 1550 checks whether the total number of strings ruined currently, denoted by n , has reached the maximum number of target ruin strings. If yes, the algorithm terminates. Otherwise, a new seed node is randomly selected to continue the ruin process at step 1520. The selection criterion requires that the new seed node be adjacent to the previous seed node, but does not belong to any routes processed by the ruin algorithm so far.

Figure 16 shows an exemplary embodiment of a recreate part of a “Ruin and Recreate” neighbourhood search method according to the present disclosure.

Step 1600 sorts the ruined nodes according to their distances from the start and finish locations, e.g. depots, and their demand, and requires that nodes far from the depots and nodes which are high in demand are inserted first. Step 1610 attempts to insert the ruined nodes, one by one, into all available routes at the best possible insertion positions, by trying all possible insertion points exhaustively and comparing the resulting route lengths. During this phase, a random probability factor is introduced to skip some insertion points. Step 1620 repeats the insertion for all ruined nodes before terminating the method.

Figure 8 shows an exemplary embodiment of a method employing “Tabu” search as metaheuristic function and a neighbourhood search function according to the present disclosure.

The principle of “Tabu” search as metaheuristic function is based on entering recently found best solutions of the search space in the solution space into a first-in, first-out memory structure of a pre-determined size called the Tabu List. The rules for entering into the Tabu List, and the size of the Tabu List, govern the extent of diversification, or the global reach, i.e. the distance of “large jumps” into new areas of the search space. In other words, a large memory size prevents more recent best solutions to be selected, thus promoting a more global exploration of the search space, and vice versa.

At the start, Step 800 prepares an initial solution S and an empty Tabu List of a pre-determined size that promotes global exploration in the solution space. Step 800 also sets the global best solution, S_{best} , to this initial solution S . Step 810 performs a search around the neighbourhood of S for a solution with a better “cost value” than S , i.e. a better or more optimal solution. The neighbourhood search function must exclude all solutions in the Tabu List in this particular iteration. Once found, this new solution replaces the current solution S .

Step 820 checks whether the new solution S is better than the global best solution S_{best} , by comparing their cost values. If S has a better cost value, then S becomes the new S_{best} at step 830. Regardless of whether S becomes the new global best solution or not, S is inserted into Tabu List at step 840, so that it would not be considered again for the next few iterations.

Step 850 checks whether conditions for terminating the iteration have been met. If met, the global best solution is output. Otherwise, the neighbourhood search function is resumed at step 810, but this time, centring on the new solution S . Possible termination conditions may be, e.g., that the cost difference between successive global best solutions has dropped below a threshold, or the maximum number of iterations have been reached, whichever occurs first.

In one embodiment of this invention, after every several search iterations, the Tabu List is pruned and the most dated solutions in the list are purged, to vary the neighbourhood search direction from global exploration to local exploration.

Figure 9 illustrates one embodiment of the invention based on a dynamic Tabu List. The functionalities of steps 900, 910, 920, 930, 940 and 950 mirror steps 800, 810, 820, 830 840 and 850 of Figure 8 respectively. At the end of every few iterations, or every iteration, the Tabu List is pruned, and the oldest solution(s) are discarded, at Step 960. This results in a similar search space reduction effect like the embodiment describes in Figure 8.

Figure 10 shows an exemplary embodiment of a method employing a metaheuristic function guiding an adaptive “Ruin and Recreate” neighbourhood search function and a local neighbourhood search function according to the present disclosure. In particular, an embodiment of the invention is shown that uses Simulated Annealing as the metaheuristic function that guides two neighbourhood search methods operating in tandem: an Adaptive Ruin and Recreate method and a Local Search method.

The functionalities of 1000, 1010, 1015, 1020, 1030, 1035, 1040, 1050, 1060 mirror steps 200, 210, 215, 220, 230, 235, 240, 250, 260 of Figure 2 respectively. However, instead of a Ruin and Recreate function, based solely on the currently best solution “S”, an Adaptive Ruin and Recreate function is employed, which takes into consideration additionally a global parameter of the metaheuristic function, e.g. the temperature parameter “T”.

Like step 230, step 1030 checks whether the temperature T is below a threshold $T_{\min}$. If the condition is true, the method terminates and returns the last solution as the final solution in step 1035. Otherwise, the process compares the temperature with another threshold T_s at step 1040. If the temperature is above T_s and further Adaptive Ruin and Recreate iterations are possible, the process performs the next “Adaptive Ruin and Recreate” iteration at step 1010, which potentially introduces large jumps in the solution space. Dependent on the (current) value of the temperature parameter “T”, these jumps can be influenced in their scale or magnitude. In other words, a relatively high temperature may allow larger jumps in the solution space than a relatively low temperature, which limits the jump distance or adjacency range.

Otherwise, if the temperature is below T_s , the process switches to a “Local Search” search function at step 1050, which introduces more refined, smaller jumps in the solution space. Such a “Local Search” search function may e.g. be hill climbing or steepest descent. Step 1040 may thus be seen as the step where the condition to switch from the first search function, here exemplarily adaptive ruin and recreate, to a second search function, here exemplarily “Local Search” is met. The solutions produced by “Local Search” is also subject to temperature-dependent processing at step 1060, just like the solutions by “Adaptive Ruin and Recreate” at step 1015.

It should be noted that, even though the above embodiments were disclosed individually, the scope of the invention extends to all technically plausible combinations of these embodiments. For example, for the Tabu Search-based embodiment illustrated in Figure 9, its neighbourhood search function at step 910 may be replaced by an adaptive Ruin and Recreate neighbourhood

search function, e.g. according to Figure 3. See e.g. Figure 10. Alternatively, the neighbourhood search function of Figure 9 may also start with a Ruin and Recreate neighbourhood search function but switch to simple local search after a plurality of iterations or when certain conditions are met.

5 Figure 4 shows an exemplary embodiment of the transportation system according to the present disclosure. In particular, Figure 4 shows an exemplary ride sharing embodiment of the present invention.

 Users 4000 submit their bookings, either individually or collectively, using their electronic devices to the booking system 4010. A booking may be made via a booking application that is
10 wirelessly accessible through mobile computing platform, like e.g. a smartphone, a smart (digital) watch, a tablet, a laptop, a short messaging service (SMS), a mobile messaging application, e.g. WhatsApp, WeChat, via a centralized booking portal, or a manual or automated voice call system.

 The booking system 4000 processes the bookings to produce structured booking data 4020 for the transport system 4140. The structured booking data 4020 may contain at least some
15 of requested pick-up location, requested drop-off location, pick-up time or drop-off time and required capacity.

 Vehicle data 4060 and driver data, in the case of manned vehicles, are registered with a vehicle control System 4070 before service operation. When the vehicles are servicing the users, they also communicate real time information with the vehicle control system 4070. The vehicle
20 control system 4070 processes vehicle and driver information into structured vehicle data 4080 for transportation system 4140. The structured vehicle data 4080 may contain at least some of a driver schedule, an average vehicle speed, a vehicle capacity and a current location.

 Before and during service operation, the routing system 4140 may regularly gather or is provided with at least some of road network data, road closure schedule and traffic data, collectively referred to as the road data 4130, from e.g. a road network and traffic service portal
25 4120 or some other instance providing such data.

 To provide the ride sharing service of the present invention, the transportation system 4140 is employed, and employs at least some of the provided data to solve the vehicle routing problem as described above. Specifically, it generates an initial solution from the structured
30 booking data 4020, the structured vehicle data 4080 and optionally the road data 4090. The initial solution is then refined over a plurality of iterations using the computer-implemented method for route calculation and optimization illustrated in Figure 2 or Figure 3, to produce a set of optimal routes. Each route data contains, at least, a linked list of nodes, vehicle info and/or driver info of each route. Each node contains, at least, offered pick-up time, offered drop-off time.

35 A subset of the above optimal route info 4030 is delivered to the booking system 4010, which is then sent to users who have a current booking 4050. Exemplarily, the sent info 4050

comprises at least an offered/rejected status, and if offered an offered pick-up time, an offered drop-off time, a vehicle registration number and optionally, the driver info.

Further subset of the above optimal route info 4030 is delivered to the vehicle control system 4070, which is then sent to the vehicles or the drivers 4110. Exemplarily, the sent info 4110 comprises at least all the nodes of the optimised route associated with the vehicle.

The above embodiment assumes that arbitrary requested pick-up and drop-off locations 4020 are legally permitted service locations. In reality, transportation laws and regulations might restrict service locations to, for example, only existing bus stops and/or taxi stands. Given a requested location, there might be a plurality of permitted locations within acceptable walking distance.

The alternative permitted locations may be selected based on a simple “shortest walking distance” criterion. Based on the chosen + alternative permitted locations, the routing engine may generate a set of routes/offers, optimised according to their cost values. Each route/offer comprises a combination of chosen and alternative permitted locations. For example, one route/offer may have a chosen permitted pick-up, but an alternative permitted drop-off. At this stage, the user may pick one of the routes (or offers), or reject them all.

Figure 5 shows a variant of the exemplary embodiment of the transportation system of Figure 4. To solve the above mentioned real-life problem, variant of the last embodiment additionally includes a node pre-processing element 5150 and a node post-processing element 5180 to compute a near-optimal route.

For each requested location 5020, the node pre-processing element 5150 selects a set of permitted locations 5160 for the transportation system 5140. The said set of permitted locations may consist of one or more than one locations. As the first variation of this embodiment, the selection may be based on a location that comprises the shortest walking distance. As the second variation, the selection may be based on a popular location according to a transit pattern. As the third variation, both locations may be selected.

For each allocated location 5170, the node post-processing element 5180 assigns an alternative allocated location, e.g. based on other criteria. As the first variation, the alternative location for each node of an optimised route are found simultaneously to satisfy a minimum route length objective. As the second variation of this embodiment, the alternative allocated location may be a popular location according to a transit pattern.

Figure 6 shows an exemplary embodiment for finding the minimum length route in the transport systems according to the present disclosure.

Areas A, B and C, represented by circles in Figure 6, are the originally requested locations by users in the transport system 4140, 5140. The permitted locations in turn are represented by triangles: A₁-A₃ are the permitted locations around A, B₁-B₂ are the permitted locations around B, and C₁-C₃ are the permitted locations around C. If the node pre-processing element 5150 would

determine pick-up locations according to the shortest walking distance criterion, then A_1 , B_1 and C_1 would be the permitted nodes 5160 output by the Transportation System 5140, because these pick-up locations are the closest to the respective requested location 5000. Thus, route $A_1 \rightarrow B_1 \rightarrow C_1$ constitutes the initial optimal route output by the Routing System in 5170.

5 The Node Post-processing element 5180 identifies all the permitted transitions between the requested locations. Based on the example of Figure 6, there are six permitted transitions from A to B: A_1B_1 , A_1B_2 , A_2B_1 , A_2B_2 , A_3B_1 and A_3B_2 . There are also six permitted transitions from B to C: B_1C_1 , B_1C_2 , B_1C_3 , B_2C_1 , B_2C_2 and B_2C_3 . In principle, there would be $6 \times 6 = 36$ possible transitions from A to C. However, some of the transitions can be pre-eliminated, e.g. based on
10 time window constraints. To find the shortest route from A to C, dynamic programming may be invoked. In this example, the shortest total route for the vehicle may be visually identified as $A_2 \rightarrow B_2 \rightarrow C_2$. As a result, A_2 , B_2 and C_2 become the alternative permitted locations that replace A_1 , B_1 and C_1 respectively.

It is understood that all the system components described herein can be hosted in a
15 plurality of computers. Further, they can either be centralised or distributed across a plurality of geographical locations. The functionalities of the components can also be handled by different computers or different parties. For example, for the vehicle control system 4070, the vehicle and driver registration can be handled by one set of computers, but the real time communication can be handled by another set of computers. As another example, for the road network and traffic
20 service portal 4120, the road network may be cached in a local database, but the traffic information may be retrieved from an external party portal.

Likewise, the transport system 4140, 5140 may be implemented in a single computing unit, further comprising the booking system 4010, 5010, the vehicle control system 4070, 5070 as well as the node pre-processing element 4150, 5150 and node post-processing element 4180,
25 5180.

In the following the technical concept of the present disclosure will be described in an alternative way, using different language. The present invention relates to a method for calculating optimal routes for a plurality of vehicles to service a plurality of bookings, comprising generating initial routes to service the bookings, improving the routes in a plurality of iterations by searching
30 for improved routes in the neighbourhood of the current routes in solution space, and designating the improved routes as the current routes, and Varying the neighbourhood search scope adaptively with each iteration. A "plurality of iterations" refers in particular to an iterative loop of a metaheuristic function, while "varying" may be understood as switching to a local search function, or a neighbourhood search function or alternatively to reducing the adjacency range of a
35 neighbourhood search function embodied as ruin and recreate. The iterations may terminate when either allocated time to search for more solutions is exceeded, a better solution than the previous solution cannot be found for a defined number of iterations, or in case the solution, e.g.

a cost function has reached a satisfactory/acceptable value, i.e. corresponds to a certain threshold. The search continues to iterate for as long as the termination conditions have not been met. The iterating method can include, but is not limited to, simulated annealing, while the neighbourhood search can include, but not limited to, ruin and recreate.

5 Further, the step of varying neighbourhood search scope may comprise reducing the adjacency range of the ruin and recreate search function. Further, the step of varying neighbourhood search scope may employ the ruin and recreate search function in some iterations and local neighbourhood search method in other, different iterations. Further, the adjacency range may be reduced according to the temperature parameter of the simulated annealing method.

10 Further, the step of switching to a local neighbourhood search method may be executed when the temperature parameter of the simulated annealing method drops below a threshold, or when the simulated annealing method no longer yields solution, whichever occurs first.

Further, the disclosure may provide a method for ride sharing, comprising, receiving a plurality of ride bookings from a booking system, where in the booking may comprises at least,

15 but not limited to, requested pick-up location, requested drop-off location, either pick-up time or drop-off time, and required capacity, may receive data for a plurality of vehicles from a vehicle control system, wherein the data comprises at least, but not limited to, driver schedule, average vehicle speed, vehicle capacity and location, may receive road data from road network and traffic service, comprising some or all of: road network, road closure schedule and traffic data, may

20 calculate optimal routes for said vehicles to service the said bookings, nay may further comprise the steps of generating initial routes to service the bookings, Improving the routes in a plurality of iterations by searching for improved routes in the neighbourhood of the current routes in solution space, and designating the improved routes as the current routes, varying the neighbourhood search scope adaptively with each iteration, dispatching a subset of the optimal route information

25 to the users directly or via the booking system, and dispatching a subset of the optimal route information to the vehicles directly or via the vehicle control system. Further, a booking system may include, but not limited to, booking applications accessible through smartphone, smart (digital) watches, tablet or laptop, short messaging service (SMS), mobile messaging applications, centralized booking portals and manual or automated voice call system that

30 optionally include application programming interfaces (API's). Further, the vehicles can be manned or autonomous, and may be terrestrial, aerial, aquatic, amphibian or triphibian. Further, the required capacity may cater to humans, animals or goods, and might additionally include special features like wheelchairs, beds, containers or holding areas. Further, the optimal route information may include, but not limited to, allocated pick-up location, allocated drop-off location,

35 pick-up time window and drop-off time window for each booking, the servicing sequence for the bookings, assigned vehicle, and driver in case of a manned vehicle.

Further, the step of calculating optimal routes may further comprise a step of pre-processing the pick-up locations into permitted pick-up locations and a step of pre-processing the drop-off locations into permitted drop-off locations. Further, the step of calculating optimal routes may further comprise a step of changing the allocated pick-up or drop-off locations of a plurality of bookings of the optimised routes. Further, the step of selecting permitted pick-up or drop-off locations may be configured to favour locations closest to the requested pick-up or drop-off locations. Further, the step of selecting permitted pick-up or drop-off locations may be configured to favour statistically popular locations. Further, the step of changing the allocated pick-up or drop-off locations may further comprise identifying a plurality of permitted locations around each allocated location, identifying all permitted transitions between said permitted locations, deriving the shortest route with said permitted transitions, and changing the allocated pick-up or drop-off locations to the permitted locations of the permitted transitions of the derived shortest route. Further, the method may comprise collecting commute preferences, including but not limited to, preferred pick-up location, preferred drop-off location, preferred pick-up time and preferred drop-off time, deriving transit pattern from said commute preferences, comprising, at least, popular locations, and changing the allocated pick-up or drop-off locations to favour popular locations.

The features disclosed in the foregoing description, or in the following claims, or in the accompanying drawings, expressed in their specific forms or in terms of a means for performing the disclosed function, or a method or process for obtaining the disclosed results, as appropriate, may, separately, or in any combination of such features, be utilised for realising the invention in diverse forms thereof.

While the invention has been described in conjunction with the exemplary embodiments described above, many equivalent modifications and variations will be apparent to those skilled in the art when given this disclosure. Accordingly, the exemplary embodiments of the invention set forth above are considered to be illustrative and not limiting. Various changes to the described embodiments may be made without departing from the scope of the invention.

For the avoidance of any doubt, any theoretical explanations provided herein are provided for the purposes of improving the understanding of a reader. The inventors do not wish to be bound by any of these theoretical explanations.

Any section headings used herein are for organizational purposes only and are not to be construed as limiting the subject matter described.

Throughout this specification, including the claims which follow, unless the context requires otherwise, the words “have”, “comprise”, and “include”, and variations such as “having”, “comprises”, “comprising”, and “including” will be understood to imply the inclusion of a stated integer or step or group of integers or steps but not the exclusion of any other integer or step or group of integers or steps.

It must be noted that, as used in the specification and the appended claims, the singular forms "a," "an," and "the" include plural referents unless the context clearly dictates otherwise. Ranges may be expressed herein as from "about" one particular value, and/or to "about" another particular value. When such a range is expressed, another embodiment includes from the one particular value and/or to the other particular value. Similarly, when values are expressed as approximations, by the use of the antecedent "about," it will be understood that the particular value forms another embodiment. The term "about" in relation to a numerical value is optional and means, for example, +/- 10%.

The words "preferred" and "preferably" are used herein refer to embodiments of the invention that may provide certain benefits under some circumstances. It is to be appreciated, however, that other embodiments may also be preferred under the same or different circumstances. The recitation of one or more preferred embodiments therefore does not mean or imply that other embodiments are not useful, and is not intended to exclude other embodiments from the scope of the disclosure, or from the scope of the claims.

Claims

1. A computer-implemented method for route calculation in a multi-vehicle environment, comprising the steps of
 - 5 a) receiving booking data corresponding to a desired trip of a user;
 - b) determining at least one first route data as current route data corresponding to the booking data;

wherein each of the at least one first route data comprises at least one first route characteristic;
 - 10 c) determining at least one subsequent route data corresponding to the booking data and being based on the current route data;

wherein each of the at least one subsequent route data comprises at least one subsequent route characteristic; and

wherein a route characteristic is indicative of at least a quality of the respective route data;
 - 15 d) comparing the at least one subsequent route characteristic with the at least one route characteristic of the current route data;
 - e) determining, based on the comparison, whether a quality of the at least one subsequent route data is increased in relation to the at least one current route data;
 - 20 f) selecting one of the current route data and the subsequent route data having the higher quality as the subsequent route data employed as route data solution or for the next iteration; and
 - g) transmitting the route data solution to at least one of a vehicle as a route solution and a user as a booking solution;
 - 25 wherein steps c) to f) are repeated until a defined exit condition is satisfied;

wherein at least one of the at least one subsequent route data is determined by employing a metaheuristic function guiding a neighbourhood search function;

wherein the neighbourhood search function is adapted to determine subsequent route data operating from the current route data; and

wherein the search scope of the neighbourhood search function is set by a parameter of the metaheuristic function.
2. The method according to the preceding claim, wherein the metaheuristic function comprises
 - 35 one of a "simulated annealing" function or a "Tabu" function.

3. The method according to at least one of the preceding claims, wherein the neighbourhood search function comprises at least one of a “ruin and recreate” search function, an adaptive “ruin and recreate” search function and a “local neighbourhood search” search function.
- 5 4. The method according to at least one of the preceding claims, wherein the metaheuristic function comprises one of a “simulated annealing” search function and a “Tabu” search function, and wherein the neighbourhood search function comprises one of
 - i) a “ruin and recreate” search function and a “local neighbourhood search” search function;
 - 10 ii) an adaptive “ruin and recreate” search function; and
 - iii) an adaptive “ruin and recreate” search function and a “local neighbourhood search” search function.
- 15 5. The method according to at least one of the preceding claims, wherein in i) or iii) the respective “ruin and recreate” search function is employed for a first number of repetitions and the “local neighbourhood search” search function is employed for a second number of repetitions.
6. The method according to at least one of the preceding claims 3 to 5, wherein setting the neighbourhood search scope by the metaheuristic function comprises reducing the adjacency range of the adaptive “ruin and recreate” search function.
- 20 7. The method according to the preceding claim, wherein the method comprises a parameter to be reduced within the progression of the repetition of at least steps c) to e), in particular a temperature parameter or a tabu list size; and wherein the adjacency range of the adaptive “ruin and recreate” search function is reduced dependent on the parameter.
- 25 8. A system for route calculation in a multi-vehicle environment, comprising
 - a communication element adapted for receiving booking data corresponding to a desired trip of a user;
 - 30 a determining element adapted for determining at least one first route data as current route data corresponding to the booking data; wherein each of the at least one first route data comprises at least one first route characteristic;
 - a determining element adapted for determining at least one subsequent route data corresponding to the booking data and being based on the current route data; wherein
 - 35 each of the at least one subsequent route data comprises at least one subsequent route characteristic; and wherein a route characteristic is indicative of at least a quality of the respective route data;

a comparing element adapted for comparing the at least one subsequent route characteristic with the at least one route characteristic of the current route data;

a determining element adapted for determining, based on the comparison, whether a quality of the at least one subsequent route data is increased in relation to the at least one current route data;

a selecting element adapted for selecting one of the current route data and the subsequent route data having the higher quality as the subsequent route data employed as route data solution or for the next iteration; and

wherein determining subsequent route data, comparing subsequent route characteristic, determining a quality increase and selecting route data with increased quality is repeated until a defined exit condition is satisfied;

wherein at least one of the at least one subsequent route data is determined by employing a metaheuristic function guiding a neighbourhood search function; and

wherein the neighbourhood search function is adapted to determine subsequent route data operating from the current route data;

wherein the search scope of the neighbourhood search function is set by the metaheuristic function.

9. The system of the preceding claim, wherein the communication element is further adapted for transmitting the route data solution to at least one of a vehicle as a route solution and a user as a booking solution.

10. A transport routing system for route scheduling in a multi-vehicle environment, the system comprising

a communication element adapted for receiving booking data from at least one user, and adapted for receiving vehicle availability data of at least one vehicle;

at least one processing element for determining at least one route data solution for the vehicle in accordance with the booking data of the at least one user;

wherein the route data solution is determined by executing the method according to at least one claims 1 to 7.

11. The transport system according to claim 10, wherein the communication element is further adapted to provide the route data solution as trip data of the determined route data to the user; and/or to provide the route data solution to the at least one vehicle.

12. The transport system according to at least one of claims 10 or 11,

wherein the route data solution comprises at least one of a requested pick-up location, an allocated pick-up location, a requested drop-off location, an allocated drop-off location, a pick-up time, a pick-up time window, a drop-off time, a drop-off time window, a required capacity, a servicing sequence, an assigned vehicle and an assigned driver; and/or

5 wherein the vehicle availability data comprises at least one of a driver schedule, an average vehicle speed, a vehicle capacity and a vehicle location;

wherein the vehicle capacity in particular comprises a number of human passengers, a number of animal passengers, a size requirement of a good to be transported, a wheelchair requirement, a bed requirement, a container size and size of a required holding
10 area.

13. The transport system according to at least one of claims 10 to 12, wherein the communication element is further adapted to

receive road data; and

15 employ the road data for determining the at least one route; in particular

wherein road data comprises at least one of road network information, traffic service information, road closure information and traffic information.

14. The transport system according to at least one of claims 10 to 13, further comprising

20 a booking system,

wherein the booking system comprises a booking application;

wherein the booking application is adapted to receive at least one booking data from a user via a mobile computing platform, a smartphone, a smart (digital) watch, a tablet, a laptop, a short messaging service (SMS), a mobile messaging application, a centralized booking
25 portal, a telephone, and a voice call system.

15. The transport system according to at least one of claims 10 to 14,

wherein determining route data further comprises pre-processing, by a pre-processing element, pick-up locations and drop-off locations into permitted pick-up locations and
30 permitted drop-off locations, in particular

wherein at least some of the permitted pick-up locations and permitted drop-off locations are associated with a weighting function; wherein the locations closer to the requested locations and/or statistically popular locations are assigned a higher weight parameter than locations further away from the requested locations, and wherein the weighing function is
35 employed when determining permitted pick-up and permitted drop-off locations.

16. The transport system according to at least one of claims 10 to 15, wherein determining route data further comprises changing the allocated pick-up and/or drop-off locations of at least one route data.
- 5 17. The transport system according to the preceding claim, wherein changing the allocated pick-up and/or drop-off locations further comprises the system being adapted to
identify a plurality of permitted locations around each allocated location;
identify all permitted transitions between said permitted locations;
determine a shortest route with said permitted transitions, and
10 change the allocated pick-up and/or drop-off locations to the permitted locations of the permitted transitions of the derived shortest route.
18. The transport system according to at least one of claims 10 to 17, further being adapted to
collect commute preferences comprising preferred pick-up location, preferred drop-off
15 location, preferred pick-up time and preferred drop-off time;
determine a transit pattern from said commute preferences, at least comprising at least one popular location; and
change an allocated pick-up or drop-off location to favour popular locations.
- 20 19. A computer program product comprising instructions which, when the program is executed by a computer, cause the computer to carry out the method according to at least one of claim 1 to 7.
- 25 20. A computer-readable medium having stored thereon the computer program product of the preceding claim.