

SERVER AND METHOD FOR FACILITATING UPDATING BATCHING SYSTEM FOR ON-DEMAND SERVICE

TECHNICAL FIELD

[0001] Various embodiments relate to a server and a method for facilitating updating a batching system for an on-demand service.

BACKGROUND

[0002] Due to development of information technology, a user (who in some contexts herein may also be referred to as a “consumer” or an “eater”) may request an on-demand service using a computing device. The on-demand service may allow the user to fulfil the user’s demand via an immediate access to goods and/or services. The user may request the on-demand service, for example, a delivery service or a transport service, using a user interface presented on the computing device.

[0003] To improve efficiency in providing the on-demand service, an on-demand service provider has introduced a batching system. The batching system may allow the on-demand service provider to batch multiple orders (also referred to as “bookings”) if at least one criterion is met, and a delivery service provider (who in some contexts herein may also be referred to as a “driver”, a “delivery partner” or a “delivery agent”) to take the batched multiple orders at a time. For example, if the multiple orders are received in a same time slot and locations relating to the multiple orders are close to each other, such multiple orders may be batched. The

batching may help the on-demand service provider improve trip-per-transit-hour (TPTH) and reduce cost-to-serve (CTS), and may also help the delivery service provider earn more.

[0004] Success metrics in batching which focus on how efficient batching is made may be a batching rate and an overall time efficiency (OTE). The batching rate may represent the number of bookings that get batched, while the OTE may represent a trip time saved through batching.

[0005] To ensure that the batching system operates optimally, as typically measured by the batching rate and the OTE, one or more batching parameters used by the batching system may need to be updated (tuned) accordingly. However, optimising the batching parameters may be an expensive trial-and-error process involving time consuming simulations and real experiments to merely cover a subset of the batching parameters over a huge search space of a combination of possible batching parameters. For example, conventional technologies in tuning the batching system, such as running simulations, conducting A/B test experiments, and applying an auto-adaptive strategy, may be time consuming. As a consequence, it may often take weeks to months to adjust the batching parameters and maintain optimal batching performance in light of ever-changing market conditions.

[0006] Therefore, there is a need to provide a solution for facilitating updating of batching systems for on-demand services.

SUMMARY

[0007] In accordance with various embodiments, there is a server for facilitating updating a batching system for an on-demand service, the server comprising: a memory for storing instructions; a communication interface configured to receive a plurality of orders for the on-demand service from a plurality of computing devices; and a processor for executing the stored

instructions and configured to: generate a plurality of candidate batches from the received orders using one or more batching parameters; evaluate each of the plurality of candidate batches using one or more predetermined batching constraints; determine, based on the evaluation of the plurality of candidate batches, one or more first candidate batches from the plurality of candidate batches that meet the one or more predetermined batching constraints; evaluate each of the one or more first candidate batches using one or more predetermined trip constraints; determine, based on the evaluation of the one or more first candidate batches, one or more second candidate batches from the one or more first candidate batches that meet the one or more predetermined trip constraints; obtain a proxy metric, using a number of orders of the one or more second candidate batches and a number of the received orders, to determine a performance of the one or more batching parameters; and input the proxy metric and the one or more batching parameters into the batching system, to update the batching system.

[0008] In some embodiments, the processor comprises a booking evaluator, and the processor is configured to: evaluate, by the booking evaluator, each of the plurality of candidate batches by checking whether each of the plurality of candidate batches meets the one or more predetermined batching constraints using a batching constraints checker; and determine, by the booking evaluator, at least a part of the plurality of candidate batches which meets the one or more predetermined batching constraints, as the one or more first candidate batches.

[0009] In some embodiments, the processor comprises a trip evaluator, and the processor is configured to: evaluate, by the trip evaluator, each of the one or more first candidate batches by checking whether each of the one or more first candidate batches meets the one or more predetermined trip constraints using a trip constraints checker; and determine, by the trip evaluator, at least a part of the one or more first candidate batches which meets the one or more predetermined trip constraints, as the one or more second candidate batches.

[0010] In some embodiments, the processor is configured to, for each first candidate batch of the one or more first candidate batches, generate, by the trip evaluator, one or more candidate trips which can be formed in each first candidate batch; evaluate, by the trip evaluator, each of the one or more candidate trips by checking whether each of the one or more candidate trips meets the one or more predetermined trip constraints; determine, by the trip evaluator, that at least one of the one or more candidate trips meets the one or more predetermined trip constraints; and determine, by the trip evaluator, the first candidate batch, as the one or more second candidate batches.

[0011] In some embodiments, the processor is configured to pass, to the trip evaluator, the at least a part of the plurality of candidate batches which meets the one or more predetermined batching constraints, and filter out at least a part of the plurality of candidate batches which does not meet the one or more predetermined batching constraints.

[0012] In some embodiments, the proxy metric includes a proxy batching rate, and the processor is configured to estimate the proxy batching rate based on the number of orders of the one or more second candidate batches with respect to the number of the received orders.

[0013] In some embodiments, the processor is configured to adjust at least one batching parameter of the batching system, based on the proxy metric and the one or more batching parameters.

[0014] In some embodiments, the one or more batching parameters include at least one of a batching interval and a batching SLA (service-level agreement).

[0015] In some embodiments, the one or more predetermined batching constraints include at least one of a vehicle capacity, a maximum distance between pick-up locations of batched orders, and a maximum distance between drop-off locations of the batched orders.

[0016] In some embodiments, the one or more predetermined trip constraints include at least one of a minimum distance efficiency of a trip of the batched orders, a minimum time efficiency of the trip of the batched orders, a pick-up SLA of the trip of the batched orders, and a drop-off SLA of the trip of the batched orders.

[0017] In accordance with various embodiments, there is a method for facilitating updating a batching system for an on-demand service, the method comprising: receiving a plurality of orders for the on-demand service from a plurality of computing devices; generating a plurality of candidate batches from the received orders using one or more batching parameters; evaluating each of the plurality of candidate batches using one or more predetermined batching constraints; determining, based on the evaluation of the plurality of candidate batches, one or more first candidate batches from the plurality of candidate batches that meet the one or more predetermined batching constraints; evaluating each of the one or more first candidate batches using one or more predetermined trip constraints; determining, based on the evaluation of the one or more first candidate batches, one or more second candidate batches from the one or more first candidate batches that meet the one or more predetermined trip constraints; obtaining a proxy metric, using a number of orders of the one or more second candidate batches and a number of the received orders, to determine a performance of the one or more batching parameters; and inputting the proxy metric and the one or more batching parameters into the batching system, to update the batching system.

[0018] In some embodiments, the method further comprises: evaluating, by a booking evaluator, each of the plurality of candidate batches by checking whether each of the plurality of candidate batches meets the one or more predetermined batching constraints using a batching constraints checker; and determining, by the booking evaluator, at least a part of the plurality

of candidate batches which meets the one or more predetermined batching constraints, as the one or more first candidate batches.

[0019] In some embodiments, the method further comprises: evaluating, by a trip evaluator, each of the one or more first candidate batches by checking whether each of the one or more first candidate batches meets the one or more predetermined trip constraints using a trip constraints checker; and determining, by the trip evaluator, at least a part of the one or more first candidate batches which meets the one or more predetermined trip constraints, as the one or more second candidate batches.

[0020] In some embodiments, the method further comprises: for each first candidate batch of the one or more first candidate batches, generating, by the trip evaluator, one or more candidate trips which can be formed in each first candidate batch; evaluating, by the trip evaluator, each of the one or more candidate trips by checking whether each of the one or more candidate trips meets the one or more predetermined trip constraints; determining, by the trip evaluator, that at least one of the one or more candidate trips meets the one or more predetermined trip constraints; and determining, by the trip evaluator, the first candidate batch, as the one or more second candidate batches.

[0021] In some embodiments, the method further comprises: passing to the trip evaluator, the at least a part of the plurality of candidate batches which meets the one or more predetermined batching constraints, and filtering out at least a part of the plurality of candidate batches which does not meet the one or more predetermined batching constraints.

[0022] In some embodiments, the proxy metric includes a proxy batching rate, and the estimating the proxy metric comprises estimating the proxy batching rate based on the number of orders of the one or more second candidate batches with respect to the number of the received orders.

[0023] In some embodiments, the method further comprises: adjusting at least one batching parameter of the batching system, based on the proxy metric and the one or more batching parameters.

[0024] In some embodiments, the one or more batching parameters include at least one of a batching interval and a batching SLA.

[0025] In some embodiments, the one or more predetermined batching constraints include at least one of a vehicle capacity, a maximum distance between pick-up locations of batched orders, and a maximum distance between drop-off locations of the batched orders.

[0026] In some embodiments, the one or more predetermined trip constraints include at least one of a minimum distance efficiency of a trip of the batched orders, a minimum time efficiency of the trip of the batched orders, a pick-up SLA of the trip of the batched orders, and a drop-off SLA of the trip of the batched orders.

[0027] According to various embodiments, a data processing apparatus configured to perform the method of any one of the above embodiments is provided.

[0028] According to various embodiments, a computer program element comprising program instructions, which, when executed by one or more processors, cause the one or more processors to perform the method of any one of the above embodiments is provided.

[0029] According to various embodiments, a computer-readable medium comprising program instructions, which, when executed by one or more processors, cause the one or more processors to perform the method of any one of the above embodiments is provided. The computer-readable medium may include a non-transitory computer-readable medium.

BRIEF DESCRIPTION OF THE DRAWINGS

[0030] The invention will be better understood with reference to the detailed description when considered in conjunction with the non-limiting examples and the accompanying drawings, in which:

- FIG. 1 illustrates an infrastructure of a system including a server for facilitating updating a batching system for an on-demand service according to various embodiments.
- FIG. 2 illustrates a block diagram of a server for facilitating updating a batching system for an on-demand service according to various embodiments.
- FIG. 3 illustrates a flowchart for a method for facilitating updating a batching system for an on-demand service according to various embodiments.
- FIG. 4 illustrates a simplified data flow diagram of a system for facilitating updating a batching system for an on-demand service according to various embodiments.
- FIG. 5 illustrates a data flow diagram of a system for facilitating updating a batching system for an on-demand service according to various embodiments.
- FIGS. 6A and 6B illustrate detailed data flow diagrams of a system for facilitating updating a batching system for an on-demand service according to various embodiments.

DETAILED DESCRIPTION

[0031] The following detailed description refers to the accompanying drawings that show, by way of illustration, specific details and embodiments in which the disclosure may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the disclosure. Other embodiments may be utilized and structural, and logical changes may be

made without departing from the scope of the disclosure. The various embodiments are not necessarily mutually exclusive, as some embodiments can be combined with one or more other embodiments to form new embodiments.

[0032] Embodiments described in the context of one of a server and a method are analogously valid for the other server and method. Similarly, embodiments described in the context of a server are analogously valid for a method, and vice-versa.

[0033] Features that are described in the context of an embodiment may correspondingly be applicable to the same or similar features in the other embodiments. Features that are described in the context of an embodiment may correspondingly be applicable to the other embodiments, even if not explicitly described in these other embodiments. Furthermore, additions and/or combinations and/or alternatives as described for a feature in the context of an embodiment may correspondingly be applicable to the same or similar feature in the other embodiments.

[0034] In the context of various embodiments, the articles “a”, “an” and “the” as used with regard to a feature or element include a reference to one or more of the features or elements.

[0035] As used herein, the term “and/or” includes any and all combinations of one or more of the associated listed items.

[0036] Throughout the description, the term “module” may be understood as an application specific integrated circuit (ASIC), an electronic circuit, a combinational logic circuit, a field programmable gate array (FPGA), a processor which executes code, other suitable hardware components which provide the described functionality, or any combination thereof. The term of “module” may include a memory which stores code executed by the processor.

[0037] In the following, embodiments will be described in detail.

[0038] FIG. 1 illustrates an infrastructure of a system 200 including a server 100 for facilitating updating a batching system for an on-demand service according to various embodiments.

[0039] As shown in FIG. 1, the system 200 may include, but is not limited to, the server 100, a database system 140, a network 150, a computing device 160, and one or more external devices 170, 180 (not shown).

[0040] In some embodiments, the on-demand service may be a service allowing a user 161 (who in some contexts herein may also be referred to as a “consumer” or an “eater”) to fulfil the user’s demand via an immediate access to items and/or services. The user 161 may request the on-demand service, for example, a transport service or an item delivery service, using a user interface presented on the computing device 160. The user 161 may make an order for the on-demand service.

[0041] In some embodiments, the network 150 may include, but is not limited to, a Local Area Network (LAN), a Wide Area Network (WAN), a Global Area Network (GAN), or any combination thereof. The network 150 may provide a wireline communication, a wireless communication, or a combination of the wireline and wireless communication between the server 100 and the computing device 160, and between the server 100 and the one or more external devices 170, 180, for example, one or more delivery service provider devices 170, and one or more item provider devices 180.

[0042] In some embodiments, the computing device 160 may be connectable to the server 100 via the network 150. In some embodiments, the computing device 160 may be arranged in data or signal communication with the server 100 via the network 150. In some embodiments, the computing device 160 may include, but is not limited to, at least one of the following: a mobile phone, a tablet computer, a laptop computer, a desktop computer, a head-mounted display and a smart watch. In some embodiments, the computing device 160 may be associated with the user 161. For example, the computing device 160 may belong to the user 161. Although not

shown, it may be appreciated that the system 200 may further include a plurality of computing devices each associated with, for example, belonging to, a plurality of users.

[0043] In some embodiments, the computing device 160 may include a location sensor. In some embodiments, the location sensor may communicate with at least one of a global positioning satellite (GPS) server, a network server, and a Wi-Fi server, to detect a location of the computing device 160. In some embodiments, the computing device 160 may generate information about the location of the computing device 160.

[0044] In some embodiments, the server 100, for example, implemented by a server computer, may include a communication interface 110, a processor 120, and a memory 130 (as will be described with reference to FIG. 2).

[0045] In some embodiments, the server 100 may communicate with the computing device 160 via the network 150. In some embodiments, the computing device 160 may receive a request (hereinafter, referred to as an “order” or a “booking”) from the user 161 for the on-demand service. The computing device 160 may send the order to the server 100 via the network 150. In some embodiments, the computing device 160 may send the information about the location of the computing device 160 to the server 100 via the network 150. The location of the computing device 160 may be considered as a location of the user 161.

[0046] In some embodiments, the system 200 may further include a database 141. In some embodiments, the database 141 may be a part of the database system 140 which may be external to the server 100. The server 100 may communicate with the database 141. In some other embodiments, although not shown, the database 141 may be implemented locally in the memory 130 of the server 100.

[0047] In some embodiments, the server 100 may communicate with the one or more external devices 170, for example, the one or more delivery service provider devices 170, via the

network 150. The one or more delivery service provider devices 170 may be associated with one or more delivery service providers 171 (who in some contexts herein may also be referred to as “drivers”, “delivery partners” or “delivery agents”) respectively. For example, the one or more delivery service provider devices 170 may belong to the one or more delivery service providers 171 respectively. The one or more delivery service providers 171 may provide a delivery service from a first location, for example, a location of a selected item provider 181a, to a second location, for example, a delivery location such as a location of the user 161.

[0048] In some embodiments, the one or more delivery service providers 171 may take two or more orders which meet at least one criterion (also referred to as “batched orders” or “batch”). In some embodiments, the server 100 may receive a plurality of orders from the plurality of computing devices. The server 100 may batch the two or more orders which meet the at least one criterion. For example, if the two or more orders are received in a same time slot (for example, a predetermined time slot (e.g. 5 minutes)) and locations relating to the two or more orders are close to each other (for example, within a predetermined distance), the server 100 may batch the two or more orders, and assign the batched orders to one delivery partner. The delivery partner may take the batched orders at a time. For example, the delivery partner may pick up a first item from a first location, for example, a location of a first item provider 181a, and pick up a second item from a third location, for example, a location of a second item provider 181b, before delivering the first item to a second location, for example, a location of a first user 161a. The delivery partner may then deliver the first item to the second location, for example, the location of the first user 161a, and deliver the second item to a fourth location, for example, a location of a second user 161b (as will be described with reference to FIG. 2).

[0049] In some embodiments, the server 100 may communicate with the one or more external devices 180, for example, the one or more item provider devices 180, via the network 150. The

one or more item provider devices 180 may be associated with one or more item providers 181 respectively. For example, the one or more item provider devices 180 may belong to the one or more item providers 181 respectively. The one or more item providers 181 may include, but are not limited to, a food provider and a goods provider, that can manufacture and/or provide items, for example, foods or goods. For example, the food provider may include, but is not limited to, a restaurant and a café. As an example, the goods provider may include, but is not limited to, a store, a market, and a supermarket. In some embodiments, the server 100 may receive the order for the item delivery service with the information about the location of the user 161, and then produce a list of items, associated with at least one item provider, which can be prepared and delivered to the user 161. In some embodiments, the server 100 may communicate with the one or more item provider devices 180 to check the one or more item providers' 181 availability. In some embodiments, the server 100 may communicate with the one or more item provider devices 180 to aggregate information including, but not limited to, a list of available items, an estimated time of preparation of each item, and an estimated price of each item, in order to produce the list of items. In some embodiments, the computing device 160 may display the list of items with the aggregated information of the at least one of the one or more item providers 181 on the user interface. In some embodiments, after the user 161 makes selections on the user interface for the order for the item delivery service, for example, by selecting the item provider 181a and the item, the server 100 may communicate with an item provider device 180a (not shown) of the selected item provider 181a to prepare the selected item.

[0050] FIG. 2 illustrates a block diagram of a server 100 for facilitating updating a batching system for an on-demand service according to various embodiments.

[0051] As shown in FIG. 2, the server 100, for example, implemented by a server computer, may include a communication interface 110, a processor 120, and a memory 130.

[0052] In some embodiments, the memory 130 (also referred to as a “database”) may store input data and/or output data temporarily or permanently. In some embodiments, the memory 130 may store program code which allows the server 100 to perform a method S300 (as will be described with reference to FIG. 3). In some embodiments, the program code may be embedded in a Software Development Kit (SDK). The memory 130 may include an internal memory of the server 100 and/or an external memory. The external memory may include, but is not limited to, an external storage medium, for example, a memory card, a flash drive, and a web storage.

[0053] In some embodiments, the communication interface 110 may allow one or more computing devices, including a computing device 160, to communicate with the processor 120 of the server 100 via a network 150, as shown in FIG. 1. In some embodiments, as shown in FIG. 1, the computing device 160 may belong to a user 161 who wants to make an order for an on-demand service. In some embodiments, the communication interface 110 may transmit signals to the computing device 160, and/or receive signals from the computing device 160 via the network 150.

[0054] In some embodiments, the communication interface 110 may allow one or more external devices 170, 180, for example, one or more delivery service provider devices 170, and one or more item provider devices 180, to communicate with the processor 120 of the server 100 via the network 150, as shown in FIG. 1. In some embodiments, the communication interface 110 may transmit signals to the one or more external devices 170, 180, and/or receive signals from the one or more external devices 170, 180, via the network 150.

[0055] In some embodiments, the communication interface 110 may receive an order for an on-demand service, for example, an item delivery service to be provided to the user 161, from the computing device 160 associated with the user 161 via the network 150. In some embodiments, the order for the item delivery service may include information about at least one selected item and at least one selected item provider providing the at least one selected item. The communication interface 110 may then send the order for the item delivery service to the processor 120.

[0056] In some embodiments, the communication interface 110 may further receive information about a location of the computing device 160 from the computing device 160 via the network 150. The communication interface 110 may then send the information about the location of the computing device 160 to the processor 120.

[0057] In some embodiments, the communication interface 110 may receive concurrently the order for the item delivery service and the information about the location of the computing device 160 from the computing device 160 via the network 150. In some other embodiments, the communication interface 110 may receive the order for the item delivery service first, and subsequently receive the information about the location of the computing device 160, from the computing device 160 via the network 150.

[0058] The processor 120 may include, but is not limited to, a microprocessor, an analogue circuit, a digital circuit, a mixed-signal circuit, a logic circuit, an integrated circuit, a Central Processing Unit (CPU), a Graphics Processing Unit (GPU), a Digital Signal Processor (DSP), a Field Programmable Gate Array (FPGA), an Application Specific Integrated Circuit (ASIC), or any combination thereof. Any other kind of implementation of the respective functions, which will be described below in further detail, may also be understood as the processor 120.

[0059] In some embodiments, the processor 120 may be connectable to the communication interface 110. In some embodiments, the processor 120 may be arranged in data or signal communication with the communication interface 110 to receive the order for the item delivery service and the information about the location of the computing device 160.

[0060] In some embodiments, the processor 120 may receive concurrently the order for the item delivery service and the information about the location of the computing device 160 from the communication interface 110. In some other embodiments, the processor 120 may receive the order for the item delivery service first, and subsequently receive the information about the location of the computing device 160 from the communication interface 110.

[0061] In some embodiments, the processor 120 may receive a request for a search for the item delivery service from the computing device 160. In some embodiments, the processor 120 may produce a list of items, associated with at least one item provider, which can be prepared and delivered to the user 161. The processor 120 may then provide the list of items to the computing device 160, so that the user 161 may select the item provider 181a and the item for the item delivery service.

[0062] In some embodiments, the user 161 may select the item provider 181a and the item from the list of items. The computing device 160 may generate information about the selected item provider 181a and the selected item, based on the user's 161 input. The processor 120 may receive the information about the selected item provider 181a and the selected item from the computing device 160, via the communication interface 110. In some embodiments, the processor 120 may then provide the information about the selected item to the selected item provider 181a for preparing the selected item, via the communication interface 110.

[0063] In some embodiments, the processor 120 may select a delivery service provider 171a from one or more delivery service providers 171 based on a geographical location of the one

or more delivery service providers 171. In some embodiments, the processor 120 may request the selected delivery service provider 171a to pick up the selected item at a geographical location of the selected item provider 181a and deliver the selected item to the user 161.

[0064] In some embodiments, the processor 120 may receive a plurality of orders from a plurality of computing devices each associated with a plurality of users via the communication interface 110, and the processor 120 may batch two or more orders which meet the at least one criterion among the plurality of orders. For example, if the two or more orders are received in a same time slot (for example, a predetermined time slot (e.g. 5 minutes)) and locations relating to the two or more orders are close to each other (for example, within a predetermined distance), the processor 120 may batch the two or more orders, and assign the batched orders to the selected delivery service provider 171a. In some embodiments, the locations relating to the two or more orders may include pick-up locations of the two or more orders (for example, locations of item providers 181 of the two or more orders) and/or delivery locations of the two or more orders.

[0065] In some embodiments, the processor 120 may determine a delivery route (hereinafter, also referred to as a “trip”) of the selected delivery service provider 171a. For example, the processor 120 may determine the delivery route of the selected delivery service provider 171a based on pick-up locations and delivery locations of the batched orders. In some embodiments, the processor 120 may provide the delivery route of the selected delivery service provider 171a to a delivery service provider device 170a associated with the selected delivery service provider 171a. In some embodiments, the processor 120 may provide the delivery route of the selected delivery service provider 171a to computing devices each associated with users who made the batched orders.

[0066] In some embodiments, the selected delivery service provider 171a may take the batched orders at a time. For example, the selected delivery service provider 171a may pick up a first item from a first location, for example, a location of a first item provider 181a, and pick up a second item from a third location, for example, a location of a second item provider 181b, before delivering the first item to a second location, for example, a location of a first user 161a. The selected delivery service provider 171a may then deliver the first item to the second location, for example, the location of the first user 161a, and deliver the second item to a fourth location, for example, a location of a second user 161b.

[0067] In some embodiments, the communication interface 110 may receive the plurality of orders for the on-demand service from the plurality of computing devices. For example, the plurality of orders may include a current order and other orders which appear while the current order is available for batching. The processor 120 may receive the plurality of orders for the on-demand service from the communication interface 110.

[0068] In some embodiments, the processor 120 may include, but is not limited to, a booking evaluator configured to generate a plurality of candidate batches, evaluate each of the plurality of candidate batches using one or more predetermined batching constraints, and determine one or more first candidate batches from the plurality of candidate batches based on the evaluation of each of the plurality of candidate batches (as will be described below). For example, the booking evaluator may be formed as a software module installed in the processor 120. As another example, the booking evaluator may be a hardware component, such as a controller or a processor, physically included in the processor 120.

[0069] In some embodiments, the processor 120, for example, the booking evaluator, may generate the plurality of candidate batches from the received orders. In some embodiments, the plurality of candidate batches may be all possible batches that can be formed out of the received

orders. For example, the all possible batches may be all combinations of the possible batches between the current order and the other orders which appear while the current order is available for batching.

[0070] In some embodiments, the processor 120, for example, the booking evaluator, may generate the plurality of candidate batches using one or more batching parameters (also referred to as “batching configuration parameters” or “batching configurations”). In some embodiments, the one or more batching parameters may include, but are not limited to, at least one of a batching interval and a batching SLA (service-level agreement). In some embodiments, the batching interval may refer to a difference between subsequent batching attempts. In some embodiments, the batching SLA may refer to a level of delivery service to inform a batching outcome expected by the user from the on-demand service provider that may attempt batching orders, laying out metrics by which that service is measured, and/or penalties and remedies (if any) in case the agreed service levels may not be achieved. In this manner, an availability of an order for a batching attempt may be dictated by the batching interval and the batching SLA. In some embodiments, the one or more batching parameters may further include at least one of an order delivery delay, and a maximum amount of time the order can be kept (also referred to as a “max pooling duration”), and the availability of the order for the batching attempt may further be dictated by the order delivery delay and the max pooling duration. In some embodiments, the one or more batching parameters may be set by an operator of an on-demand service platform (for example, an engineer, a data scientist, etc.).

[0071] In some embodiments, the processor 120, for example, the booking evaluator, may evaluate each of the plurality of candidate batches using the one or more predetermined batching constraints. In some embodiments, the processor 120, for example, the booking

evaluator, may evaluate each of the plurality of candidate batches, by checking each of the plurality of candidate batches using the one or more predetermined batching constraints.

[0072] In some embodiments, the processor 120, for example, the booking evaluator, may evaluate each of the plurality of candidate batches by checking whether each of the plurality of candidate batches meets the one or more predetermined batching constraints using a batching constraints checker. In some embodiments, the batching constraints checker may be included in the processor 120. For example, the batching constraints checker may be formed as a software module installed in the processor 120. As another example, the batching constraints checker may be a hardware component, such as a controller or a processor, physically included in the processor 120. In some other embodiments, the batching constraints checker may be external to the server 100. For example, the batching constraints checker may be formed as a software module installed in another server or processor which is external to the server 100. As another example, the batching constraints checker may be a hardware component, such as a controller or a processor, physically included in another server or processor which is external to the server 100.

[0073] In some embodiments, the processor 120, for example, the booking evaluator, may determine one or more first candidate batches from the plurality of candidate batches based on the evaluation of each of the plurality of candidate batches. In some embodiments, the processor 120, for example, the booking evaluator, may determine at least a part of the plurality of candidate batches which meets the one or more predetermined batching constraints, as the one or more first candidate batches.

[0074] In some embodiments, the one or more predetermined batching constraints may include, but are not limited to, at least one of a vehicle capacity, a maximum distance between

pick-up locations of batched orders, and a maximum distance between drop-off locations of the batched orders.

[0075] In some embodiments, the processor 120, for example, the booking evaluator, may pass the at least a part of the plurality of candidate batches which meets the one or more predetermined batching constraints (feasible batches) to a trip evaluator, to proceed with a next process. The processor 120, for example, the booking evaluator, may filter out at least a part of the plurality of candidate batches which does not meet the one or more predetermined batching constraints (infeasible batches), not to pass to the trip evaluator, not to proceed with the next process. In this manner, the processor 120, for example, the booking evaluator, may allow only feasible batches to be evaluated by the trip evaluator, by pruning a number of infeasible batches in the process.

[0076] In some embodiments, the processor 120 may include, but is not limited to, the trip evaluator configured to evaluate each of the one or more first candidate batches using one or more predetermined trip constraints, and determine one or more second candidate batches from the one or more first candidate batches based on the evaluation of each of the one or more first candidate batches (as will be described below). For example, the trip evaluator may be formed as a software module installed in the processor 120. As another example, the trip evaluator may be a hardware component, such as a controller or a processor, physically included in the processor 120. In some embodiments, the trip evaluator may be included in the booking evaluator. In some other embodiments, the trip evaluator may be external to the booking evaluator.

[0077] In some embodiments, the processor 120, for example, the trip evaluator, may evaluate each of the one or more first candidate batches using the one or more predetermined trip constraints. In some embodiments, the processor 120, for example, the trip evaluator, may

evaluate each of the one or more first candidate batches, by checking each of the one or more first candidate batches using the one or more predetermined trip constraints.

[0078] In some embodiments, the processor 120, for example, the trip evaluator, may evaluate, each of the one or more first candidate batches by checking whether each of the one or more first candidate batches meets the one or more predetermined trip constraints using a trip constraints checker. In some embodiments, the trip constraints checker may be included in the processor 120. For example, the trip constraints checker may be formed as a software module installed in the processor 120. As another example, the trip constraints checker may be a hardware component, such as a controller or a processor, physically included in the processor 120. In some other embodiments, the trip constraints checker may be external to the server 100. For example, the trip constraints checker may be formed as a software module installed in another server or processor which is external to the server 100. As another example, the trip constraints checker may be a hardware component, such as a controller or a processor, physically included in another server or processor which is external to the server 100.

[0079] In some embodiments, the processor 120, for example, the trip evaluator, may determine one or more second candidate batches from the one or more first candidate batches based on the evaluation of each of the one or more first candidate batches. In some embodiments, the processor 120, for example, the trip evaluator, may determine at least a part of the one or more first candidate batches which meets the one or more predetermined trip constraints, as the one or more second candidate batches.

[0080] In some embodiments, the one or more predetermined trip constraints may include, but are not limited to, at least one of a minimum distance efficiency of a trip of the batched orders, a minimum time efficiency of the trip of the batched orders, a pick-up SLA of the trip of the batched orders, and a drop-off SLA of the trip of the batched orders.

[0081] In some embodiments, for each first candidate batch of the one or more first candidate batches, the processor 120, for example, the trip evaluator, may perform the evaluation and determination. In some embodiments, for each first candidate batch, the processor 120, for example, the trip evaluator, may generate one or more candidate trips which can be formed in each first candidate batch. In some embodiments, the one or more candidate trips may be all possible trips that can be formed out of the orders in the same batch (for example, that first candidate batch). For example, the all possible trips may be all combinations of the possible trips that can be formed out of the orders in the same batch. In some embodiments, for each first candidate batch, the processor 120, for example, the trip evaluator, may evaluate each of the one or more candidate trips by checking whether each of the one or more candidate trips meets the one or more predetermined trip constraints. In some embodiments, for each first candidate batch, the processor 120, for example, the trip evaluator, may determine that at least one of the one or more candidate trips meets the one or more predetermined trip constraints. In some embodiments, for each first candidate batch, the processor 120, for example, the trip evaluator, may determine the first candidate batch (for example, that first candidate batch), as the one or more second candidate batches. In this manner, the processor 120, for example, the trip evaluator, may check whether each of the one or more candidate trips is feasible or not. If there is at least one feasible candidate trip for a particular batch, then the batch may be deemed batch-able. In some embodiments, a batch-able order may refer to an order that can be batched with at least one other order with a valid batched trip.

[0082] In some embodiments, the processor 120 may include, but is not limited to, a metric extractor configured to obtain a proxy metric (also referred to as a “batching quality metric”, a “batching related metric” or a “batching metric”), using a number of orders of the one or more second candidate batches and a number of the received orders (as will be described below). For

example, the metric extractor may be formed as a software module installed in the processor 120. As another example, the metric extractor may be a hardware component, such as a controller or a processor, physically included in the processor 120. In some embodiments, the metric extractor may be included in the booking evaluator. In some other embodiments, the metric extractor may be external to the booking evaluator.

[0083] In some embodiments, the processor 120, for example, a metric extractor, may obtain the proxy metric including a booking-level batching proxy metric. The booking-level batching proxy metric may relate to a batching outcome, including, but not limited to, at least one of a proxy batching rate (also referred to as a “batching rate”) and whether a particular booking is batch-able or not. In some embodiments, the processor 120, for example, the metric extractor, may obtain the proxy metric using the number of orders of the one or more second candidate batches and the number of the received orders. In some embodiments, the proxy batching rate may be obtained based on the number of orders of the one or more second candidate batches with respect to the number of the received orders. For example, the proxy batching rate may be calculated by dividing the number of orders of the one or more second candidate batches into the number of the received orders. In some embodiments, the processor 120, for example, the metric extractor, and/or the batching system may use the obtained proxy metric to determine a performance of the one or more batching parameters which were used to generate the plurality of candidate batches.

[0084] In some embodiments, the processor 120 may extract the one or more batching parameters which were used to generate the plurality of candidate batches. the processor 120 may input the obtained proxy metric and the one or more batching parameters into the batching system, to update the batching system. In some embodiments, the batching system may be updated using the obtained proxy metric and the one or more batching parameters which were

used to generate the plurality of candidate batches, to improve the batching system. For example, the processor 120 may adjust a current batching parameter of the batching system, based on the obtained proxy metric and the one or more batching parameters which were used to generate the plurality of candidate batches.

[0085] In some embodiments, in view that the obtained proxy metric is used to determine the performance of the one or more batching parameters which were used to generate the plurality of candidate batches, the processor 120 and/or the batching system may determine, based on the obtained proxy metric, whether to update the batching system using the one or more batching parameters. For example, if the obtained proxy metric is equal to or greater than a predetermined threshold, the processor 120 and/or the batching system may determine that the one or more batching parameters are optimal, and determine to update the batching system using the one or more batching parameters which were used to generate the plurality of candidate batches, to improve the batching system. In this manner, the batching system may be updated using the one or more batching parameters which were used to generate the plurality of candidate batches, and the updated batching system may be used to batch new orders. As another example, if the obtained proxy metric is less than the predetermined threshold, the processor 120 and/or the batching system may determine that the one or more batching parameters are not optimal, and determine not to update the batching system using the one or more batching parameters which were used to generate the plurality of candidate batches.

[0086] As described above, the various embodiments may prune a search space for finding optimal batching parameters to be reasonably small. This may allow other optimisation approaches, for example, an A/B test experiment and an auto-adaptive strategy, to focus on a few promising batching parameters to optimise. In this manner, it may take much less time to

adjust the batching parameters to meet ever-changing market conditions, ultimately keeping batching performance at its peak most of the time.

[0087] FIG. 3 illustrates a flowchart for a method S300 for facilitating updating a batching system for an on-demand service according to various embodiments. According to various embodiments, the method S300 for facilitating updating the batching system for an on-demand service may be provided.

[0088] In some embodiments, the method S300 may include a step S301 of receiving a plurality of orders for the on-demand service from a plurality of computing devices. For example, the plurality of orders may include a current order and other orders which appear while the current order is available for batching.

[0089] In some embodiments, the method S300 may include a step S302 of generating a plurality of candidate batches from the received orders using one or more batching parameters. In some embodiments, the one or more batching parameters may include, but are not limited to, at least one of a batching interval and a batching SLA.

[0090] In some embodiments, the method S300 may include a step S303 of evaluating each of the plurality of candidate batches using one or more predetermined batching constraints. In some embodiments, each of the plurality of candidate batches may be evaluated, by checking whether each of the plurality of candidate batches meets the one or more predetermined batching constraints.

[0091] In some embodiments, the method S300 may include a step S304 of determining, based on the evaluation of the plurality of candidate batches, one or more first candidate batches from the plurality of candidate batches that meet the one or more predetermined batching constraints. In some embodiments, at least a part of the plurality of candidate batches, which meets the one

or more predetermined batching constraints, may be determined as the one or more first candidate batches.

[0092] In some embodiments, the method S300 may include a step S305 of evaluating each of the one or more first candidate batches using one or more predetermined trip constraints. In some embodiments, each of the one or more first candidate batches may be evaluated, by checking whether each of the one or more first candidate batches meets the one or more predetermined trip constraints.

[0093] In some embodiments, the method S300 may include a step S306 of determining, based on the evaluation of the one or more first candidate batches, one or more second candidate batches from the one or more first candidate batches that meet the one or more predetermined trip constraints. In some embodiments, at least a part of the one or more first candidate batches, which meets the one or more predetermined trip constraints, may be determined as the one or more second candidate batches.

[0094] In some embodiments, the method S300 may include a step S307 of obtaining a proxy metric, using a number of orders of the one or more second candidate batches and a number of the received orders, to determine a performance of the one or more batching parameters.

[0095] In some embodiments, the method S300 may include a step S308 of inputting the proxy metric and the one or more batching parameters into the batching system, to update the batching system.

[0096] FIG. 4 illustrates a simplified data flow diagram of a system 200 for facilitating updating a batching system for an on-demand service according to various embodiments. In some embodiments, the server 100 described with reference to FIG. 2 may perform operations relating to FIG. 4.

[0097] In some embodiments, the system 200, for example, the server 100, may include a batching proxy model (BPM) 230. The BPM 230 may be a fast and cheap-to-evaluate model that is capable of estimating at least one proxy metric 250 (also referred to as “batching quality metric”, “batching related metric” or “batching metric”), for example, a batching rate (also referred to as a “proxy batching rate”), considering one or more batching parameters 220 (also referred to as “batching configuration parameters” or “batching configurations”). Unlike a conventional batching simulation that imitates almost identically real batching processes, the BPM 230 may imitate a simplified version of essential behaviours in the batching processes.

[0098] In practice, the batching processes may contain four basic mechanisms as follows: a divide-and-conquer mechanism, a constraints check mechanism, a solution search mechanism, and a quality check mechanism. The divide-and-conquer mechanism may divide big problems into smaller ones. The constraints check mechanism which usually works in tandem with the solution search mechanism may find a feasible and optimal solution. The quality check mechanism may ensure that the solution found is good enough to be sent for allocation. The conventional batching simulation may imitate all these batching processes, while the BPM 230 simplifies these batching processes into two processes as follows: the constraint check mechanism and the solution search mechanism, by incorporating the quality check mechanism into the constraint check mechanism, and ignoring the divide-and-conquer mechanism. Furthermore, the solution search strategy that is used in the BPM 230 may be simpler than the one in the conventional batching simulation, as the solution search strategy may be designed to find a feasible solution instead of an optimal solution, which is much faster to achieve.

[0099] As shown in FIG. 4, there may be two sets of inputs to the BPM 230. The inputs may include the batching configuration 220 and bookings data 210 (also referred to as “order data” or “demand data”). Given the inputs, the BPM 230 may provide a proxy metric 250 (also

referred to as a “batching quality metric”, a “batching related metric” or a “batching metric”), for example, the batching rate, as an output. In some embodiments, a user of an on-demand service platform, such as an operator of the on-demand service platform (for example, an engineer, a data scientist, etc.), may set various parameters under the batching configurations, to control batching process behaviours while processing the incoming demand data 210 and form (generate) batches from the incoming demand data 210. The various parameters which may be set by the operator may include, but are not limited to, batching quality constraints, user experience constraints, etc. Similarly, the BPM 230 may need to consider both the batching configuration 220 and the demand data 210 to estimate the proxy metric 250. In some embodiments, the various parameters may be extracted by the metric extractor 233 to inform the operator about an input to the system 200, and the values of the various parameters may be the same as the one inputted to the system 200, which is categorised as the one or more batching configurations (the one or more batching parameters). This information (for example, the one or more batching parameters) along with the proxy metric 250 generated by BPM 230 may be used to improve the batching system in the on-demand service platform.

[00100] FIG. 5 illustrates a data flow diagram of a system 200 for facilitating updating a batching system for an on-demand service according to various embodiments. In some embodiments, the server 100 described with reference to FIG. 2 may perform operations relating to FIG. 5.

[00101] In some embodiments, the system 200, for example, the server 100, may include a batching proxy model (BPM) 230. In some embodiments, the BPM 230 may include a plurality of components to approximate the proxy metric 250, as follows:

- a) a bookings evaluator 231, configured to evaluate all combinations of batches that can be formed out of the bookings (also referred to as “orders”);

- b) a batch constraints checker 241, configured to ensure that only feasible batches are evaluated by the trip evaluator 232, by pruning infeasible batches in the process;
- c) a trip evaluator 232, configured to evaluate all combinations of trips that can be formed out of the bookings in a batch;
- d) a trip constraints checker 242, configured to check whether a trip is feasible or not – if there is at least one feasible trip for a particular batch, the batch is deemed batch-able; and
- e) a metric extractor 233, configured to measure how many batches are feasible.

[00102] FIGS. 6A and 6B illustrate detailed data flow diagrams of a system 200 for facilitating updating a batching system for an on-demand service according to various embodiments. FIGS. 6A and 6B illustrate how booking and their potential pairs are selected for batch-ability computation. In some embodiments, the server 100 described with reference to FIG. 2 may perform operations relating to FIGS. 6A and 6B.

[00103] In some embodiments, the system 200, for example, the server 100, may include a batching proxy model (BPM) 230. In some embodiments, the BPM 230 may include, but is not limited to, a booking evaluator 231, a trip evaluator 232, and a metric extractor 233. In some embodiments, as shown in FIG. 6A, the trip evaluator 232 and the metric extractor 233 may be included in the booking evaluator 231. Although not shown, in some other embodiments, the trip evaluator 232 and the metric extractor 233 may be external to the booking evaluator 231.

[00104] In some embodiments, the BPM 230 may receive bookings data 210 (also referred to as “order data”) relating to a plurality of bookings (also referred to as “a plurality of orders”) from a plurality of computing devices.

[00105] In some embodiments, the booking evaluator 231 may start with looping over all bookings one by one. For each booking, for example, a current booking 211 (also referred to

as a “current order”), the booking evaluator 231 may form all possible batches (also referred to as “candidate batches”) that can be formed between the current booking 211 and other bookings 212 (also referred to as a “other orders”) which appear while the current booking 211 is available for batching (S231). In some embodiments, the availability of a booking for a batching attempt may be dictated by one or more batching parameters 221. For example, the one or more batching parameters 221 may include, but are not limited to, a batching interval and a batching SLA.

[00106] In some embodiments, once the candidate batches are formed, a batching constraints checker 241 may prune infeasible batches. In some embodiments, the batching constraints may include, but are not limited to, at least one of a vehicle capacity 241a, a maximum distance between pick-up locations of batched orders 241b, and a maximum distance between drop-off locations of the batched orders 241c. Advantageously, unnecessary computation on the infeasible batches may be avoided.

[00107] In some embodiments, as shown in FIG. 6B, the batching constraints checker 241 may use at least one batching parameter 222 to prune the infeasible batches. In some embodiments, the at least one batching parameter 222 may include, but is not limited to, a vehicle capacity (relating to the vehicle capacity 241a of the batching constraints), a maximum distance between pick-up locations of batched orders (relating to the maximum distance between pick-up locations of batched orders 241b of the batching constraints), and a maximum distance between drop-off locations of the batched orders (relating to the maximum distance between drop-off locations of the batched orders 241c of the batching constraints).

[00108] In some embodiments, thereafter, the feasible batches may be passed to the trip evaluator 232 to further evaluate whether the trip, which may include a pick-up/drop-off steps sequence and an arrival time at each step, is feasible or not. In some embodiments, the trip

evaluator 232 may loop over all trips one by one. In some embodiments, the trip evaluator 232 may form all possible trips (also referred to as “possible routes”) for each of the feasible batches received from the booking evaluator 231 (S232a). The trip evaluator 232 may use a trip constraint checker 242 to check whether any of possible (feasible) trips from a batch meet all the trip constraints (S232b). In some embodiments, the trip constraints may include, but are not limited to, at least one of a minimum distance efficiency of a trip of the batched orders 242a, a minimum time efficiency of the trip of the batched orders 242b, a pick-up SLA of the trip of the batched orders 242c, and a drop-off SLA of the trip of the batched orders 242d. In some embodiments, when forming a trip, the trip evaluator 232 may assume a best-case scenario for routing, for example, which may be straight-line distance/duration used to check the SLA constraints.

[00109] In some embodiments, as shown in FIG. 6B, the trip constraint checker 242 may use at least one batching parameter 223 to check whether any of possible (feasible) trips from a batch meet all the trip constraints. In some embodiments, the at least one batching parameter 223 may include, but is not limited to, a minimum distance efficiency of a trip of the batched orders (relating to the minimum distance efficiency of a trip of the batched orders 242a of the trip constraints), a minimum time efficiency of the trip of the batched orders (relating to the minimum time efficiency of the trip of the batched orders 242b of the trip constraints), and a drop-off SLA of the trip of the batched orders and a maximum drop-off delay (relating to the pick-up SLA of the trip of the batched orders 242c and the drop-off SLA of the trip of the batched orders 242d of the trip constraints).

[00110] In some embodiments, the metric extractor 233 may extract a batching proxy metric, for example, a booking-level batching proxy metric, from feasible batches including any of possible (feasible) trips meeting all the trip constraints. For example, the metric extractor 233

may get the booking-level batching proxy metric from aggregated metrics of feasible pairs and routes (S233). In some embodiments, the booking-level batching proxy metrics may relate to a batching outcome, including, for example, at least one of a proxy batching rate and whether a particular booking is batch-able or not. If there is at least one possible (feasible) trip for a particular batch, the batch is deemed batch-able, and the booking-level batching proxy metric may be obtained in relation to the batch.

[00111] In some embodiments, the booking-level batching proxy metric may be output as a resulting metric 250 (also referred to as a “proxy metric”, a “batching quality metric”, a “batching related metric” or a “batching metric”). For example, the resulting metric 250 may include the proxy batching rate. In some embodiments, the proxy batching rate may be calculated using the following mathematical equation:

$$\text{Proxy Batching Rate} = \frac{\text{Number of batchable bookings}}{\text{Number of all bookings}}$$

where a batch-able booking may refer to a booking that can be batched with at least one other booking with a valid batched trip.

[00112] For example, the proxy batching rate is “1.0”, if all bookings are batch-able with at least one other booking during a batching attempt, while the proxy batching rate is “0.0” if no booking can be batched with any other booking.

[00113] As described above, the batching proxy model (BPM) according to the various embodiments may enhance existing strategies employed to optimise batching parameters, by guiding the search for the optimal batching parameters using OTE (overall time efficiency) estimates over a huge search space. The BPM may run a lower fidelity simulation, for example, including (only) essential processes in batching to estimate the batching related metrics (for example, the batching rate). The BPM may simplify complex processes behind batching, for example, divide-and-conquer, and solve vehicle routing problems, apply quality checks by

simply checking whether an order can be batched with any other order, and then estimate a potential batching quality of the possible batches.

[00114] Advantageously, the BPM may allow to evaluate a wide range of the batching parameters much faster than other approaches with a reasonably accurate estimate. For example, running an A/B test experiment with multiple treatment cases may typically require multi-days efforts, but with a simulation, speeds that are approximately 10x faster than in the A/B test experiment may be observed. In addition, the observations from initial experiments may show that the BPM may speed that process up by another 10x than the simulation. In other words, the BPM may evaluate 100x more parameters, compared to running the A/B test experiment.

[00115] While the disclosure has been particularly shown and described with reference to specific embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the appended claims. The scope of the invention is thus indicated by the appended claims and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced.

CLAIMS

1. A server for estimating a proxy batching rate for an on-demand service, the server comprising:

a memory for storing instructions;

a communication interface configured to receive order data corresponding to a plurality of bookings for the on-demand service from a plurality of computing devices; and

a processor for executing the stored instructions and configured to:

generate candidate batches from the plurality of bookings based on one or more batching parameters relating to availability of bookings for batching;

evaluate the candidate batches using one or more batching constraints to identify feasible batches;

for each feasible batch, determine whether at least one candidate trip satisfies one or more trip constraints;

identify bookings that can be batched with a valid batched trip, based on the determination; and

obtain the proxy batching rate based on a number of the bookings that can be batched with respect to a total number of the plurality of bookings.

2. The server according to claim 1, wherein the proxy batching rate is obtained without executing a full batching simulation or route optimisation.

3. The server according to claim 1 or claim 2, wherein the processor is further configured to determine a performance of the one or more batching parameters based on the proxy batching rate.
4. The server according to claim 3, wherein the processor is further configured to adjust at least one batching parameter, based on the proxy batching rate and the one or more batching parameters.
5. The server according to any one of claims 1 to 4, wherein the processor is further configured to:
 - evaluate each of the candidate batches by checking whether each of the candidate batches meets the one or more batching constraints; and
 - identify at least a part of the candidate batches which meets the one or more batching constraints, as the feasible batches.
6. The server according to claim 5, wherein the processor is configured to pass the at least a part of the candidate batches which meets the one or more batching constraints to a next process, and filter out at least a part of the candidate batches which does not meet the one or more batching constraints.
7. The server according to any one of claims 1 to 6 wherein the one or more batching parameters include at least one of a batching interval and a batching SLA (service-level agreement).

8. The server according to any one of claims 1 to 7, wherein the one or more batching constraints include at least one of a vehicle capacity, a maximum distance between pick-up locations of batched bookings, and a maximum distance between drop-off locations of the batched bookings.

9. The server according to claim 8, wherein the one or more trip constraints include at least one of a minimum distance efficiency of a trip of the batched bookings, a minimum time efficiency of the trip of the batched bookings, a pick-up SLA of the trip of the batched bookings, and a drop-off SLA of the trip of the batched bookings.

10. A method for estimating a proxy batching rate for an on-demand service, the method comprising:

receiving order data corresponding to a plurality of bookings for the on-demand service from a plurality of computing devices;

generating candidate batches from the plurality of bookings based on one or more batching parameters relating to availability of bookings for batching;

evaluating the candidate batches using one or more batching constraints to identify feasible batches;

for each feasible batch, determining whether at least one candidate trip satisfies one or more trip constraints;

identifying bookings that can be batched with a valid batched trip, based on the determination; and

obtaining the proxy batching rate based on a number of the bookings that can be batched with respect to a total number of the plurality of bookings.

11. The method according to claim 10, wherein the proxy batching rate is obtained without executing a full batching simulation or route optimisation.
12. The method according to claim 10 or claim 11, further comprising: determining a performance of the one or more batching parameters based on the proxy batching rate.
13. The method according to claim 12, further comprising: adjusting at least one batching parameter, based on the proxy batching rate and the one or more batching parameters.
14. The method according to any one of claims 10 to 13, further comprising:
 - evaluating each of the candidate batches by checking whether each of the candidate batches meets the one or more batching constraints; and
 - identifying at least a part of the candidate batches which meets the one or more batching constraints, as the feasible batches.
15. The method according to claim 14, further comprising: passing the at least a part of the candidate batches which meets the one or more batching constraints to a next process, and filtering out at least a part of the candidate batches which does not meet the one or more batching constraints.