

Method, Apparatus and Electronic Device for Optimizing Picking Path

Cross-reference of Related Applications

The present application claims the priority of the Chinese patent application CN202410332947.6, titled "Method, Apparatus and Electronic Device for Optimizing Picking Path", filed on March 22, 2024. The entire content of this application is incorporated herein by reference.

Technical Field

The present application belongs to the technical field of logistics, and particularly relates to a method, an apparatus and an electronic device for optimizing a picking path.

Background Art

The planning of the picking path in a warehouse is the core link of outbound, which refers to a process of taking out products from a storage area (also known as storage location) of the warehouse, and carrying out operations such as collection, classification, and packing according to an order. In the picking link, a picker can take out the products required in the order one by one according to the planned picking path, and carry out inspection, counting, marking, etc., and then place them in the corresponding packing position. Usually, the picking link of the warehousing operation is one of the links that take the most time and concentrate the most manpower in the whole operation process. However, the traditional picking path planning methods often rely on manual experience and rules, resulting in low efficiency and accuracy.

Summary

The present application provides a method, an apparatus and an electronic device for optimizing a picking path, which are used to solve the technical problems of low efficiency and accuracy of the picking path planning method in the prior art.

In the first aspect, an embodiment of the present application provides a method for optimizing a picking path, comprising: obtaining a storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises a location of a storage area corresponding to the products to be picked, locations of the aisles within the storage area, and storage locations within the aisles; generating at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch; and optimizing the picking path of the picking task according to a picking strategy.

In the above solution, after obtaining the storage location layout information corresponding to the orders to be planned, multiple picking tasks can be generated for each order batch according to the above storage location layout information, and the picking paths of the above picking tasks can be optimized according to the picking strategy, so as to obtain the optimal picking path. By using the method for optimizing the picking path provided in the embodiment of the present application, since the picking path can be optimized automatically according to the storage location layout

information, compared with the prior art in which the picking path is determined manually, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, optimizing the picking path of the picking task according to the picking strategy comprises: optimizing the in-aisle path of the picking task according to the picking strategy to obtain an in-aisle picking sequence within the aisle; and optimizing an inter-aisle path of the picking task according to the in-aisle picking sequence and a shortest path rule to obtain an in-storage area picking sequence within the storage area. In the above solution, in the process of optimizing the picking path of the picking task, the in-aisle path can be optimized first, and then the inter-aisle path can be further optimized according to the optimized in-aisle path, so as to ensure the optimality of the inter-aisle picking path on the basis of ensuring the accuracy of the in-aisle picking path planning.

In an optional implementation manner, the picking strategy comprises at least one of the following: selecting the picking path with the shortest distance within the aisle; selecting the picking path with the fewest times of climbing ladders within the aisle; selecting the picking path with the fewest times of squatting within the aisle; and selecting the picking path with the fewest times of turning around within the aisle. In the above solution, a picking strategy for inter-aisle path optimization can be formulated based on the picking habits of pickers, so as to obtain a picking path with higher picking efficiency.

In an optional implementation manner, optimizing the inter-aisle path of the picking task according to the in-aisle picking sequence and the shortest path rule to obtain the in-storage area picking sequence within the storage area comprises: constructing a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule; and determining the in-storage area picking sequence by solving the source-free shortest path problem. In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

In an optional implementation manner, constructing a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule comprises: determining two nodes corresponding to the aisle according to the in-aisle picking sequence corresponding to the aisle, wherein the nodes represent the shortest path within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and a picking sequence corresponding to the two nodes is opposite; and determining the in-storage area picking sequence by solving the source-free shortest path problem comprises: calculating an adjacent distance between the nodes of two adjacent aisles; determining multiple target nodes that minimize a distance from the first aisle to the last aisle according to the adjacent distance, wherein each aisle corresponds to one target node; and determining the in-storage area picking sequence according to the multiple target nodes. In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

In an optional implementation manner, generating multiple picking tasks according to the storage location layout information comprises: for an order batch, using a greedy algorithm to generate the multiple picking tasks according to the storage location layout information. In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, using a greedy algorithm to generate the multiple picking tasks according to the storage location layout information comprises: determining a seed product, wherein the seed product is the product to be picked that has the most common aisles with other products to be picked among the products to be picked that have not been arranged, and has the smallest number of aisles, columns, and layers; initializing a current picking task with the seed product; determining an insertion product from the products to be picked that have not been arranged using a greedy rule, wherein the insertion product is the product to be picked that has the lowest insertion cost when inserted into the current picking task; and inserting the insertion product into the current picking task until number of the products to be picked in the current picking task is greater than a number threshold. In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the insertion cost represents an increase in the picking path after inserting the product to be picked.

In an optional implementation manner, generating at least one order batch according to the storage location layout information comprises: determining whether the order to be planned comprises multiple products to be picked; and if the order to be planned comprises multiple products to be picked, dividing the multiple orders to be picked into multiple order batches according to the storage location layout information and batching conditions; otherwise, generating one order batch according to all the orders to be planned. In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

In an optional implementation manner, dividing the multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions comprises: determining the order batch that covers the smallest number of storage areas according to constraint conditions; wherein the constraint conditions comprise: one order to be picked is assigned to one order batch; number of the orders to be picked in one order batch is not greater than the maximum number of orders to be picked that can be assigned to the order batch; number of the products to be picked for one storage area in one order batch is not greater than a total number of orders to be picked. In the above solution, the division method of order batches that minimizes the number of storage areas covered by one order batch can be solved, so that the optimal solution for order batch division can be obtained, thereby improving the picking efficiency.

In an optional implementation manner, dividing the multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions comprises: constructing a product location set corresponding to the orders to be picked, wherein values in the product location set represent storage area numbers where the products to be picked in the orders to be picked are located; sorting the values in the product location set and concatenating them into a string; sorting the multiple orders to be picked according to the strings corresponding to the multiple orders to be picked; and dividing the sorted orders to be picked according to the maximum number of the orders to be picked that can be assigned to the order batch. In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

In the second aspect, an embodiment of the present application provides an apparatus for optimizing a picking path, comprising: an acquisition module, configured to obtain a storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises a location of a storage area corresponding to the products to be picked, locations of aisles within the storage area, and storage locations within the aisles; a generation module, configured to generate at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch; and an optimization module, configured to optimize the picking path of each picking task according to a picking strategy.

In the above solution, after obtaining the storage location layout information corresponding to the orders to be planned, multiple picking tasks can be generated for each order batch according to the above storage location layout information, and the picking paths of the above picking tasks can be optimized according to the picking strategy, so as to obtain the optimal picking path. By using the method for optimizing the picking path provided in the embodiment of the present application, since the picking path can be optimized automatically according to the storage location layout information, compared with the prior art in which the picking path is determined manually, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the optimization module is specifically configured to: optimize the in-aisle path of the picking task according to the picking strategy to obtain the in-aisle picking sequence within the aisle; optimize the inter-aisle path of the picking task according to the in-aisle picking sequence and the shortest path rule to obtain the in-storage area picking sequence within the storage area. In the above solution, in the process of optimizing the picking path of each picking task, the in-aisle path can be optimized first, and then the inter-aisle path can be further optimized according to the optimized in-aisle path, so as to ensure the optimality of the inter-aisle picking path on the basis of ensuring the accuracy of the in-aisle picking path planning.

In an optional implementation manner, the picking strategy comprises at least one of the following: selecting the picking path with the shortest distance within the aisle; selecting the picking path with the fewest times of climbing ladders within the

aisle; selecting the picking path with the fewest times of squatting within the aisle; and selecting the picking path with the fewest times of turning around within the aisle. In the above solution, a picking strategy for inter-aisle path optimization can be formulated based on the picking habits of pickers, so as to obtain a picking path with higher picking efficiency.

In an optional implementation manner, the optimization module is specifically configured to: construct a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule; and determine the in-storage area picking sequence by solving the source-free shortest path problem. In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

In an optional implementation manner, the optimization module is further configured to: determine two nodes corresponding to an aisle according to the in-aisle picking sequence corresponding to the aisle, wherein the nodes represent the shortest path within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and the picking sequences corresponding to the two nodes are opposite; calculate the adjacent distance between the nodes of two adjacent aisles; determine multiple target nodes that minimize the distance from the first aisle to the last aisle according to the adjacent distance, wherein each aisle corresponds to one target node; and determine the in-storage area picking sequence according to the multiple target nodes. In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

In an optional implementation manner, the generation module is specifically configured to: for an order batch, use a greedy algorithm to generate the multiple picking tasks according to the storage location layout information. In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the generation module is further configured to: determine a seed product, wherein the seed product is the product to be picked that has the most common aisles with other products to be picked among the products to be picked that have not been arranged, and has the smallest number of aisles, columns, and layers; initialize the current picking task with the seed product; determine an insertion product from the products to be picked that have not been arranged using a greedy rule, wherein the insertion product is the product to be picked that has the lowest insertion cost when inserted into the current picking task; and insert the insertion product into the current picking task until the number of products to be picked in the current picking task is greater than a number threshold. In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the insertion cost represents the increase in the picking path after inserting the product to be picked.

In an optional implementation manner, the generation module is specifically configured to: determine whether the order to be planned comprises multiple products to be picked; and if the order to be planned comprises multiple products to be picked, divide the multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions; otherwise, generate one order batch according to all the orders to be planned. In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

In an optional implementation manner, the generation module is specifically configured to: determine the order batch that covers the smallest number of storage areas according to the constraint conditions; wherein the constraint conditions comprise: one order to be picked is assigned to one order batch; the number of orders to be picked in one order batch is not greater than the maximum number of orders to be picked that can be assigned to the order batch; the number of products to be picked for one storage area in one order batch is not greater than the total number of orders to be picked. In the above solution, the division method of order batches that minimizes the number of storage areas covered by one order batch can be solved, so that the optimal solution for order batch division can be obtained, thereby improving the picking efficiency.

In an optional implementation manner, the generation module is specifically configured to: construct a product location set corresponding to the orders to be picked, wherein the values in the product location set represent the storage area numbers where the products to be picked in the orders to be picked are located; sort the values in the product location set and concatenate them into a string; sort the multiple orders to be picked according to the strings corresponding to the multiple orders to be picked; and divide the sorted orders to be picked according to the maximum number of orders to be picked that can be assigned to the order batch. In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

In the third aspect, an embodiment of the present application provides an electronic device, comprising: a processor, a memory and a bus; wherein the processor and the memory communicate with each other through the bus; the memory stores computer program instructions executable by the processor, and the processor can execute the method for optimizing a picking path as described in the first aspect by invoking the computer program instructions.

In the fourth aspect, an embodiment of the present application provides a computer-readable storage medium, wherein the computer-readable storage medium stores computer program instructions, and when the computer program instructions

are run by a computer, the computer is enabled to execute the method for optimizing a picking path as described in the first aspect.

Other features and advantages of the present application will be set forth in the subsequent specification, and in part will become obvious from the specification, or can be understood by implementing the embodiments of the present application. The objectives and other advantages of the present application can be realized and obtained through the structures particularly pointed out in the written specification, claims and drawings.

Brief Description of Drawings

In order to illustrate the technical solutions of the embodiments of the present application more clearly, the following briefly introduces the figures that need to be used in the embodiments of the present application. It should be understood that the following figures only show some embodiments of the present application, and therefore should not be regarded as a limitation of the scope. For those of ordinary skill in the art, without creative efforts, other related figures can also be obtained according to these figures.

Figure 1 is a flowchart of a method for optimizing a picking path provided in an embodiment of the present application;

Figure 2 is a schematic diagram of a high-rise shelf type three-dimensional warehouse provided in an embodiment of the present application;

Figure 3 is a schematic diagram of storage locations corresponding to the products to be picked provided in an embodiment of the present application;

Figure 4 is a schematic diagram of a picking sequence of the products to be picked provided in an embodiment of the present application;

Figure 5 is a schematic diagram of an inter-aisle path optimization method provided in an embodiment of the present application;

Figure 6 is a schematic diagram of constructing a source-dependent shortest path problem provided in an embodiment of the present application;

Figure 7 is a flowchart of a greedy algorithm provided in an embodiment of the present application;

Figure 8 is a structural block diagram of a system for optimizing a picking path provided in an embodiment of the present application;

Figure 9 is a structural block diagram of an apparatus for optimizing a picking path provided in an embodiment of the present application; and

Figure 10 is a structural block diagram of an electronic device provided in an embodiment of the present application.

Detailed Description of Embodiments

The technical solutions in the embodiments of the present application will be clearly and completely described below in conjunction with the accompanying drawings in the embodiments of the present application.

It should be noted that the term "multiple" in the embodiments of the present invention refers to two or more. In view of this, "multiple" in the embodiments of the present invention can also be understood as "at least two". "And/or" describes the association relationship of associated objects, indicating that there can be three relationships. For example, A and/or B can mean: A exists alone, A and B exist simultaneously, and B exists alone. In addition, the character "/" generally indicates an "or" relationship between the associated objects before and after it, unless otherwise specified.

The problem of picking path planning can be defined as follows: Given a set of picking storage locations, determine the access sequence and path of the storage locations to achieve the goal of minimizing the distance of the picking path. Therefore, an embodiment of the present application provides a method for optimizing a picking path to automatically achieve the above goal. Please refer to Figure 1. Figure 1 is a flowchart of a method for optimizing a picking path provided in an embodiment of the present application. The method for optimizing the picking path can comprise the following steps:

Step S101: obtain the storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises the location of the storage area corresponding to the products to be picked, the location of the aisles within the storage area, and storage locations within the aisles.

Step S102: generate at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch.

Step S103: optimize the picking path of the picking task according to the picking strategy.

Specifically, in the above step S101, multiple orders to be planned refer to multiple orders that need to be picked together currently. As an implementation manner, the above multiple orders to be planned can be called a wave. A wave is a group of orders to be processed generated according to certain rules by combining several orders together in warehousing to improve the efficiency of the warehouse and reduce wrong and missing shipments. The orders in the same wave have the same attributes and processing rules.

It should be noted that the embodiment of the present application does not specifically limit the specific implementation manner of determining a wave (i.e., multiple orders to be planned). Those skilled in the art can make appropriate adjustments according to the actual situation. For example, orders within a certain period of time can be determined as a wave according to the time sequence at regular intervals; or, orders with a fixed quantity can be determined as a wave according to the order quantity; or, a wave can be determined according to the user's requirements.

Taking the determination of a wave according to the user's requirements as an example, a wave generation request refers to a request from on-site wave grouping personnel or the system to screen out an order set and create a wave according to preset wave grouping conditions. In specific implementation, when the user needs to create an order wave combination, the user can input (or set) the wave grouping conditions into the computer device, and the computer device will respond to the wave generation request and obtain the order information in the current order pool and the real-time inventory information of the warehouse.

Herein, each order to be planned can comprise one or more products to be picked. As an implementation manner, in the scenario of combined outbound shipment, an order to be planned comprises multiple products to be picked, and each product to be picked can be referred to as a detail. As another implementation mode, in the scenario of transaction-based outbound shipment, each order to be planned only comprises one detail (that is, it only comprises one product to be picked).

The storage location layout information refers to the location of the products to be picked in the orders to be planned in the warehouse. The storage location layout information can comprise the location of the storage area corresponding to the products to be picked, the location of the aisles within the storage area, and storage locations within the aisles.

Please refer to Figure 2. Figure 2 is a schematic diagram of a high-rise shelf type three-dimensional warehouse provided in an embodiment of the present application. Figure 2 shows Aisle 09, Aisle 10, Aisle 11 and Aisle 12 in the AD storage area. Among them, Aisle 09, Aisle 10, Aisle 11 and Aisle 12 all have both Sides A and B, in which Side A is composed of odd-numbered columns, Side B is composed of even-numbered columns, and each column has seven layers. It can be understood that a storage location is located at a certain layer of a certain column in a certain aisle of a certain storage area in the warehouse. For example, the storage location AD090503 in Figure 2 corresponds to the AD storage area, Aisle 09, Column 5, and the third layer.

In the warehouse shown in Figure 2, if the first column is picked first and then the second column is picked, one turn is required; while the first column and the third column can generally be picked simultaneously without turning or walking horizontally. The third and fourth layers have the highest picking efficiency. For the first and second layers, squatting is required, and for the fifth layer and above, they are high-level storage locations and ladders need to be used. After finishing the picking in Aisle 09, when entering Aisle 10, it is possible to "go forward" and bypass the largest column (Column 16), or "go back" and bypass the smallest column (Column 1), which needs to be determined according to the storage locations to be picked in the two aisles.

It should be noted that the method for optimizing the picking path provided in the embodiment of the present application is not limited to the warehouse shown in Figure 2. Figure 2 is only an example provided in the embodiment of the present application, and those skilled in the art can make appropriate adjustments to the specific implementation manner of the warehouse according to the actual situation.

In addition, the embodiment of the present application does not specifically limit the specific implementation manner of obtaining the storage location layout information corresponding to multiple orders to be planned. Those skilled in the art can make appropriate adjustments according to the actual situation. For example, the

storage location layout information sent by external devices can be received; or, the storage location layout information stored locally or in the cloud can be queried, etc.

In the above step S102, an order batch is a collection of combined outbound shipment orders that need to be sorted together. At least one order batch will be generated in one wave, and there will be at least one order to be planned under one order batch. Among them, generally, the maximum number of orders to be planned corresponds to the capacity of the subsequent in-warehouse sorting vehicle.

In the embodiment of the present application, each order to be planned is only included in one order batch.

It should be noted that the embodiment of the present application does not specifically limit the specific implementation manner of generating at least one order batch according to the storage location layout information. Those skilled in the art can make appropriate adjustments according to the actual situation. For example, the orders to be planned corresponding to the products to be picked within a certain range in the warehouse can be divided into one order batch; or, the order batches can be divided based on minimizing the number of storage areas covered by one order batch, etc.

A picking task refers to a work task of picking products generated according to customer orders, outbound instructions and other related information in the warehousing logistics system. It usually comprises the operation of taking the products off the shelves, so as to meet the demand of completing the outbound of products within the shortest time and at the lowest cost.

In the embodiment of the present application, for each order batch, the picking task corresponding to the order batch can be generated according to the storage location layout information obtained in the above step S101. Among them, the above picking task is used to pick the products to be picked in the order batch.

It should be noted that the embodiments of the present application do not specifically limit the specific implementation manner of generating picking tasks according to the storage location layout information. Those skilled in the art can make appropriate adjustments according to the actual situation. For example, picking tasks can be generated based on the picking habits of pickers; or, a greedy algorithm can be used to generate picking tasks, and so on.

In the above step S103, the picking path refers to the walking sequence and the path passed within the warehouse when completing the picking task corresponding to an order batch. In the embodiments of the present application, the picking path of the picking task can be optimized according to the pre-determined picking strategy, so as to obtain the picking path corresponding to the picking task.

It should be noted that the embodiments of the present application do not specifically limit the specific implementation manner of the picking strategy. The purposes of path optimization can comprise minimizing the picking path, minimizing the number of times the picker climbs a ladder, or minimizing the number of times the picker squats, etc. Therefore, the specific implementation manner of the picking strategy can be determined according to the actual situation and the specific purpose of path optimization.

As an implementation manner, by configuring parameters, two modes of mixed high-low picking and separated high-low picking can be realized. Among them, high-level refers to the storage locations that require the use of ladders for picking, while low-level refers to the storage locations that do not require the use of ladders for picking. In the separated high-low picking mode, the low-level storage locations of all aisles can be picked first, and then the high-level storage locations of all aisles can be picked. In the mixed high-low picking mode, the high-level and low-level storage locations of each aisle (even each column) are picked together, and it may be necessary to go up and down the ladder frequently.

It can be understood that, in the embodiments of the present application, the optimization of the picking path is to optimize the sequence of the picking paths of the products to be picked within the same storage area. Therefore, if a picking task spans multiple storage areas (in the scenario of combined outbound shipment), it can be grouped according to the storage areas, the path of each storage area can be optimized separately, and then combined finally.

In the above solution, after obtaining the storage location layout information corresponding to the orders to be planned, multiple picking tasks can be generated for each order batch according to the above storage location layout information, and the picking paths of the above picking tasks can be optimized according to the picking strategy, so as to obtain the optimal picking path. By using the method for optimizing the picking path provided in the embodiment of the present application, since the picking path can be optimized automatically according to the storage location layout information, compared with the prior art in which the picking path is determined manually, the efficiency and accuracy of the picking path planning can be improved.

Furthermore, on the basis of the above embodiment, the above step S103 can specifically comprise the following steps:

Step 1): optimize the in-aisle path of the picking task according to the picking strategy to obtain the in-aisle picking sequence within the aisle.

Step 2): optimize the inter-aisle path of the picking task according to the in-aisle picking sequence and the shortest path rule to obtain the in-storage area picking sequence within each storage area.

Specifically, as can be seen from the warehouse shown in Figure 2, pickers will first pick products within the aisle during the picking process. After picking the products in the current aisle, they can move to another aisle to pick products. Therefore, in the process of optimizing the picking path, the in-aisle path can be optimized first, and then the inter-aisle path can be optimized.

In the above step 1), the optimization of the in-aisle picking path requires taking into account the operation habits of the pickers. For example, a Z-shaped picking route can be designed within the aisle; or, two adjacent columns can be picked at one time without moving (which can be achieved by configuring the column offset), etc.

As an implementation manner, the picking strategy can comprise: selecting the picking path with the shortest distance within the aisle, selecting the picking path with the fewest times of climbing ladders within the aisle, selecting the picking path with the

fewest times of squatting within the aisle, and selecting the picking path with the fewest times of turning around within the aisle.

As another implementation manner, the picking strategy can comprise: cyclically picking products in the following sequence: middle-level storage locations, adjacent storage locations, low-level storage locations, opposite-side storage locations, and high-level storage locations. Among them, the middle-level storage locations refer to the storage locations that the picker can just reach by stretching out his hand, the adjacent storage locations refer to the storage locations that the picker can reach without moving, the low-level storage locations refer to the storage locations that the picker can reach without the help of a ladder, the opposite-side storage locations refer to the storage locations that the picker can reach by turning around, and the high-level storage locations refer to the storage locations that the picker can reach only with the help of a ladder.

For example, assume that the storage locations of the products to be picked on Side A and Side B of Aisle 15 are shown in Figure 3, in which, Layers 1-4 are low-level storage locations, and Layers 5-7 are high-level storage locations for which a ladder is required for picking. In the separated high-low picking mode, the low-level locations (i.e., Layers 4-1) are picked first, and then the high-level locations (i.e., Layers 5-7) are picked.

According to the picking habits, the layer scanning sequence of the low-level storage locations is from high to low (i.e., from Layer 4 to Layer 1), and the layer scanning sequence of the high-level storage locations is from low to high (i.e., from Layer 5 to Layer 7). There are two column scanning sequences, from small to large (i.e., from left to right) and from large to small, which are related to the entry column of this aisle. In Aisle 15 shown in the above figure, assuming that entering this aisle starts from the left, that is, Column 1, the column scanning order is from small to large.

Based on the above picking strategy, the picking sequence of the low-level storage locations is: (A,4,1)→(A,3,3)→(B,3,4)→(B,2,2)→(B,4,6)→(B,2,6)→(A,2,5)→....., which means picking the storage locations (4,1) and (3,3) on Side A in sequence. At this time, Columns 1 and 3 on Side A have been picked, and then turn back to pick the storage locations (3,4) and (2,2) in Columns 2 and 4 on Side B, and so on. Similarly, the picking sequence of the high-level storage locations is: (A,5,21)→(A,6,19)→(B,7,22)→(B,6,18)→(A,5,15)→.....

It can be understood that the above picking sequence can minimize the number of turns as much as possible.

It should be noted that each storage location within the aisle can be abstracted into an adjacency matrix model. According to the model, the storage location can be quickly located, and traversing this matrix represents how the picking path is generated. The matrix can be used to represent the characteristics of business rules, such as high-low level picking, turning-back mode, etc. These characteristics can better help pickers plan the path. Therefore, the in-aisle picking sequence shown in Figure 3 can be represented in the matrix shown in Figure 4.

In the above step 2), the inter-aisle path of each picking task can be optimized according to the in-aisle picking sequence within the aisle and the shortest path rule

to obtain the in-storage area picking sequence within each storage area. The specific implementation manner of the above inter-aisle path optimization will be introduced in subsequent embodiments of the present application.

In the above solution, during the process of optimizing the picking path for each picking task, the in-aisle path can be optimized first, and then the inter-aisle path can be further optimized based on the optimized in-aisle path. In this way, on the basis of ensuring the accuracy of the in-aisle picking path planning, the optimal inter-aisle picking path can be guaranteed.

Further, on the basis of the above embodiment, the above step of optimizing the inter-aisle path specifically comprises the following steps:

Step 1): construct a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule.

Step 2): determine the in-storage area picking sequence by solving the source-free shortest path problem.

Specifically, the inter-aisle path can be optimized by constructing and solving the source-free shortest path problem, so as to determine the in-storage area picking sequence. It should be noted that the embodiments of the present application do not specifically limit the specific implementation manner of constructing and solving the source-free shortest path problem. Those skilled in the art can make appropriate adjustments according to the actual situation.

In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

Further, on the basis of the above embodiment, the above step of constructing a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule specifically comprises the following steps:

Step 1): determine two nodes corresponding to an aisle according to the in-aisle picking sequence corresponding to the aisle, wherein the nodes represent the shortest path within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and the picking sequences corresponding to the two nodes are opposite.

Correspondingly, the above step of determining the in-storage area picking sequence by solving the source-free shortest path problem specifically comprises the following steps:

Step 2): calculate the adjacent distance between the nodes of two adjacent aisles.

Step 3): determine multiple target nodes that minimize the distance from the first aisle to the last aisle according to the adjacent distance, wherein each aisle corresponds to one target node.

Step 4): determine the in-storage area picking sequence according to the multiple target nodes.

Specifically, in the above step 1), the picking sequence of each aisle can be abstracted into two nodes. In this way, the abstracted nodes represent the shortest path within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and the picking sequences corresponding to the two nodes are opposite.

For example, please refer to the second aisle in Figure 5. There are 3 storage locations to be picked in the aisle: 3, 4, and 5. Then, there are two access sequences for these three storage locations: 3-4-5 or 5-4-3. The sequence 3-4-5 in Figure 5 can be abstracted as node 2F, and the sequence 5-4-3 in Figure 5 can be abstracted as node 2B.

It can be understood that the walking mode within the abstracted nodes is fixed, so the length of the picking path is a fixed value, and the two nodes abstracted from each aisle are fully connected to the two nodes of its adjacent aisles.

Therefore, as an implementation manner, a source-free shortest path problem can be constructed. As another implementation manner, two virtual nodes O (start node) and D (end node) can also be added to construct a source-dependent shortest path problem to determine the shortest inter-aisle path. Taking the construction of a source-dependent shortest path problem as an example, please refer to Figure 6. Figure 6 is a schematic diagram of constructing a source-dependent shortest path problem provided in an embodiment of the present application. By solving the shortest path from O to D, the optimal walking mode between every two adjacent aisles, that is, whether to turn back, can be automatically obtained.

Herein, the embodiments of the present application do not specifically limit the specific implementation manner of solving the shortest path problem. Those skilled in the art can make appropriate adjustments according to the actual situation. For example, the above steps 2)-4) can be used to solve the shortest path problem; or, algorithms such as dynamic programming, Dijkstra's algorithm, or binary tree traversal can be used to solve the shortest path problem.

In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

Further, on the basis of the above embodiment, the following introduces the specific implementation manner of generating picking tasks. The above step S102 can specifically comprise the following steps:

For an order batch, use a greedy algorithm to generate multiple picking tasks according to the storage location layout information.

Specifically, picking task decomposition means splitting all the products to be picked within the same order batch into several picking tasks. In the embodiments of the present application, the above purpose can be achieved by using a greedy algorithm. It should be noted that the embodiments of the present application do not

specifically limit the specific implementation manner of using a greedy algorithm to generate multiple picking tasks. Those skilled in the art can make appropriate adjustments according to the actual situation.

The following takes an example to introduce the specific implementation manner of a greedy algorithm provided in the embodiment of the present application. Please refer to Figure 7. Figure 7 is a flowchart of a greedy algorithm provided in the embodiment of the present application. The above step of using the greedy algorithm to generate multiple picking tasks according to the storage location layout information can specifically comprise the following steps:

Step 1): determine a seed product, wherein the seed product is the product to be picked that has the most common aisles with other products to be picked among the products to be picked that have not been assigned, and has the smallest number of aisles, columns, and layers.

Step 2): initialize the current picking task with the seed product.

Step 3): determine an insertion product from the products to be picked that have not been assigned using a greedy rule, wherein the insertion product is the product to be picked that has the lowest insertion cost when inserted into the current picking task.

Step 4): insert the insertion product into the current picking task until the number of products to be picked in the current picking task is greater than the number threshold.

Herein, the insertion cost in the above greedy algorithm represents the increment of the picking path after inserting the product to be picked, and it can be used to estimate the costs of crossing storage areas, crossing aisles, crossing columns, and crossing high and low levels.

In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

Further, on the basis of the above embodiment, the following introduces the specific implementation manner of order batching. The above step S102 can specifically comprise the following steps:

Step 1): determine whether the order to be planned comprises multiple products to be picked.

Step 2): if the order to be planned comprises multiple products to be picked, divide the multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions; otherwise, generate one order batch according to all the orders to be planned.

Specifically, in the above step 1), order batching means that for the scenario of combined outbound shipment, multiple products to be picked in the same order to be planned are required to be summarized in the same order batch, which is convenient for subsequent secondary sorting; while for the scenario of transaction-based outbound shipment, since there is only one order batch, there is no need for order batching. Therefore, it can be determined whether the order to be planned comprises multiple products to be picked.

In the above step 2), it can be considered that the goal of order batching is to minimize the number of storage areas covered by each batch, and there are two constraint conditions: 1) multiple products to be picked in the same order need to be in the same batch; 2) there is an upper limit on the maximum number of orders in each batch.

It should be noted that the embodiments of the present application do not specifically limit the specific implementation manner of order batching. Those skilled in the art can make appropriate adjustments according to the actual situation. For example, an optimal method can be used for order batching; or, a heuristic method can be used for order batching, and so on.

In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

Further, on the basis of the above embodiment, the following will introduce the specific implementation of using the optimal method for order batching. The above step of dividing multiple orders to be picked into multiple order batches according to the storage location layout information and batching conditions can specifically comprise the following steps:

Determine the order batch that covers the smallest number of storage areas according to the constraint conditions; wherein the constraint conditions comprise: one order to be picked is assigned to one order batch; the number of orders to be picked in one order batch is not greater than the maximum number of orders to be picked that can be assigned to the order batch; the number of products to be picked for one storage area in one order batch is not greater than the total number of orders to be picked. As an implementation manner, the above step of determining the order batches that cover the minimum number of storage areas according to the constraint conditions can specifically comprise the following steps:

Determine the multiple order batches using the following formula:

$$\min \sum_j \sum_a y_{aj};$$

wherein the constraint conditions are:

$$\begin{aligned} \sum_j x_{ij} &= 1, \forall i; \\ \sum_i x_{ij} &\leq M, \forall j; \\ \sum_i s_{ia} x_{ij} &\leq n y_{aj}, \forall a, j; \end{aligned}$$

where j represents the order batch, a represents the storage area, $y_{aj} = 1$ indicates that the order batch j covers the storage area a , i represents the order to be

picked, $x_{ij} = 1$ indicates that the order to be picked i is assigned to the order batch j , M represents the maximum number of orders to be picked that can be assigned to each order batch, and s_{ia} indicates that there are products to be picked in the storage area a for the order to be picked i .

In the above solution, the division method of order batches that minimizes the number of storage areas covered by one order batch can be solved, so that the optimal solution for order batch division can be obtained, thereby improving the picking efficiency.

Further, on the basis of the above embodiment, the following will introduce the specific implementation manner of using the heuristic method for order batching. The above step of dividing multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions can specifically comprise the following steps:

Step 1): construct a product location set corresponding to the orders to be picked, wherein the values in the product location set represent the storage area numbers where the products to be picked in the orders to be picked are located.

Step 2): sort the values in the product location set and concatenate them into a string, and sort the multiple orders to be picked according to the strings corresponding to the multiple orders to be picked.

Step 3): divide the sorted orders to be picked according to the maximum number of orders to be picked that can be assigned to the order batch.

Further, on the basis of the above embodiment, during the order batching process, the optimal method can be used for order batching first. If the set threshold of the solution time is exceeded, the heuristic method can be used for order batching.

The structural block diagram of a picking path optimization system provided by an embodiment of the present application. The picking path optimization system may comprise a server, a wireless local area network, and a handheld mobile terminal. Please refer to Figure 8. Figure 8 is a structural block diagram of a server provided by an embodiment of the present application. The server 800 may comprise a mapping unit 801, a query unit 802, a calculation unit 803, and a storage unit 804.

Specifically, the mapping unit 801 is configured to generate a warehouse layout plan according to the arrangement of warehouse shelves, and then calculate information such as the floor where the storage area is located, the layers where the storage locations that require the use of ladders for picking in each aisle within the storage area are located, and the maximum column of each aisle. The query unit 802 is configured to select orders and pre-occupy inventory according to manually-screened rules, and to find the storage location layout information of the selected orders to be planned from the pre-stored data. The calculation unit 803 is configured to sequentially perform order batching, picking task decomposition, and picking path planning for the selected orders to be planned (that is, it is configured to execute the picking path optimization method provided in the embodiments of the present application). The storage unit 804 is configured to store the planned picking tasks for subsequent pushing to pickers.

In the embodiment of the present application, the optimal order combination and picking path under the current wave can be efficiently calculated, reducing the operation cost. The optimal solutions for the batching scheme and the inter-aisle turning-back scheme can be calculated. No additional parameters need to be configured, and pickers do not need to have strong business qualities and systematic learning abilities. Picking habits such as the number of squats and turns are considered, which better meets the needs of users and improves the efficiency of warehouse sorting operations. Moreover, there is no need to calculate data that is difficult to collect, such as the spatial location relationships within the storage area and the walking distances between any two storage locations, which endows it with high implementability and popularization value.

Please refer to Figure 9. Figure 9 is a structural block diagram of an apparatus for optimizing a picking path provided in an embodiment of the present application. The apparatus 900 for optimizing a picking path comprises an acquisition module 901, configured to obtain the storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises the location of the storage area corresponding to the products to be picked, the location of the aisles within the storage area, and storage locations within the aisles; a generation module 902, configured to generate at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch; and an optimization module 903, configured to optimize the picking path of the picking task according to a picking strategy.

In the above solution, after obtaining the storage location layout information corresponding to the orders to be planned, multiple picking tasks can be generated for each order batch according to the above storage location layout information, and the picking paths of the above picking tasks can be optimized according to the picking strategy, so as to obtain the optimal picking path. By using the method for optimizing the picking path provided in the embodiment of the present application, since the picking path can be optimized automatically according to the storage location layout information, compared with the prior art in which the picking path is determined manually, the efficiency and accuracy of the picking path planning can be improved.

Further, on the basis of the above embodiment, the optimization module 903 is specifically configured to: optimize the in-aisle path of the picking task according to the picking strategy to obtain the in-aisle picking sequence within the aisle; and optimize the inter-aisle path of the picking task according to the in-aisle picking sequence and the shortest path rule to obtain the in-storage area picking sequence within the storage area.

In the above solution, during the process of optimizing the picking path for each picking task, the in-aisle path can be optimized first, and then the inter-aisle path can be further optimized based on the optimized in-aisle path. In this way, on the basis of ensuring the accuracy of the in-aisle picking path planning, the optimal inter-aisle picking path can be guaranteed.

Further, on the basis of the above embodiment, the picking strategy comprises at least one of the following: selecting the picking path with the shortest distance within

the aisle, selecting the picking path with the fewest times of climbing ladders within the aisle, selecting the picking path with the fewest times of squatting within the aisle, and selecting the picking path with the fewest times of turning around within the aisle.

In the above solution, a picking strategy for inter-aisle path optimization can be formulated based on the picking habits of pickers, so as to obtain a picking path with higher picking efficiency.

Further, on the basis of the above embodiment, the optimization module 903 is specifically configured to: construct a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule; and determine the in-storage area picking sequence by solving the source-free shortest path problem.

In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

In an optional implementation manner, the optimization module 903 is further configured to: determine two nodes corresponding to an aisle according to the in-aisle picking sequence corresponding to the aisle, wherein the nodes represent the shortest path within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and the picking sequences corresponding to the two nodes are opposite; calculate the adjacent distance between the nodes of two adjacent aisles; determine multiple target nodes that minimize the distance from the first aisle to the last aisle according to the adjacent distance, wherein each aisle corresponds to one target node; and determine the in-storage area picking sequence according to the multiple target nodes.

In the above solution, by abstracting the in-aisle picking sequence into two nodes and calculating the distance between the nodes, the optimal inter-aisle path can be found. Since no other parameters need to be configured as described above, the optimal solution of the access sequence of each aisle can be directly found, thereby improving the efficiency and accuracy of the picking path planning.

Further, on the basis of the above embodiment, the generation module 902 is specifically configured to: for an order batch, use a greedy algorithm to generate the multiple picking tasks according to the storage location layout information.

In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the generation module 902 is further configured to: determine a seed product, wherein the seed product is the product to be picked that has the most common aisles with other products to be picked among the products to be picked that have not been assigned, and has the smallest number of aisles, columns, and layers; initialize the current picking task with the seed product; determine an insertion product from the products to be picked that have not been assigned using a greedy rule, wherein the insertion product is the product to be picked that has the lowest insertion cost when inserted into the current picking task; and insert

the insertion product into the current picking task until the number of products to be picked in the current picking task is greater than a number threshold.

In the above solution, a greedy algorithm can be used to generate picking tasks. Since the greedy algorithm has a low time complexity, the efficiency and accuracy of the picking path planning can be improved.

In an optional implementation manner, the insertion cost represents the increment of the picking path after inserting the product to be picked.

Further, on the basis of the above embodiment, the generation module 902 is specifically configured to: determine whether the order to be planned comprises multiple products to be picked; if the order to be planned comprises multiple products to be picked, divide the multiple orders to be picked into multiple order batches according to the storage location layout information and the batching conditions; otherwise, generate one order batch according to all the orders to be planned.

In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

Further, on the basis of the above embodiment, the generation module 902 is specifically configured to: determine the order batches that cover the minimum number of storage areas according to the constraint conditions; wherein the constraint conditions comprise: one order to be picked is assigned to one order batch; the number of orders to be picked in one order batch is not greater than the maximum number of orders to be picked that can be assigned to the order batch; the number of products to be picked for one storage area in one order batch is not greater than the total number of orders to be picked.

In the above solution, the division method of order batches that minimizes the number of storage areas covered by one order batch can be solved, so that the optimal solution for order batch division can be obtained, thereby improving the picking efficiency.

In an optional implementation manner, the generation module 902 is further configured to: determine the multiple order batches using the following formula:

$$\min \sum_j \sum_a y_{aj} ;$$

wherein the constraint conditions are:

$$\begin{aligned} \sum_j x_{ij} &= 1, \forall i; \\ \sum_i x_{ij} &\leq M, \forall j; \\ \sum_i s_{ia} x_{ij} &\leq n y_{aj}, \forall a, j; \end{aligned}$$

where j represents the order batch, a represents the storage area, $y_{aj} = 1$ indicates that the order batch j covers the storage area a , i represents the order to be picked, $x_{ij} = 1$ indicates that the order to be picked i is assigned to the order batch j , M represents the maximum number of orders to be picked that can be assigned to each order batch, and s_{ia} indicates that there are products to be picked in the storage area a for the order to be picked i .

In the above solution, the division method of order batches that minimizes the number of storage areas covered by one order batch can be solved, so that the optimal solution for order batch division can be obtained, thereby improving the picking efficiency.

In an optional implementation manner, the generation module 902 is specifically configured to: construct a product location set corresponding to the orders to be picked, wherein the values in the product location set represent the storage area numbers where the products to be picked in the orders to be picked are located; sort the values in the product location set and concatenate them into a string; sort the multiple orders to be picked according to the strings corresponding to the multiple orders to be picked; and divide the sorted orders to be picked according to the maximum number of orders to be picked that can be assigned to the order batch.

In the above solution, when the order to be planned comprises multiple products to be picked, the multiple orders to be picked can be divided into order batches. Thus, on the basis of ensuring that multiple products to be picked in the same order to be planned are in the same order batch, the storage area covered by one order batch is minimized, and thus a picking path with higher picking efficiency can be obtained.

Please refer to Figure 10. Figure 10 is a structural block diagram of an electronic device provided in an embodiment of the present application. The electronic device 1000 comprises at least one processor 1001, at least one communication interface 1002, at least one memory 1003, and at least one communication bus 1004. The communication bus 1004 is configured to realize the connection and communication among these components. The communication interface 1002 is configured to communicate with other node devices for signaling or data. The memory 1003 stores machine-readable instructions executable by the processor 1001. When the electronic device 1000 is running, the processor 1001 communicates with the memory 1003 through the communication bus 1004, and when the machine-readable instructions are called by the processor 1001, the above method for optimizing the picking path is executed.

For example, the processor 1001 in the embodiment of the present application reads a computer program from the memory 1003 through the communication bus

1004 and executes the computer program to implement the following method. Step S101: obtain the storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises the location of the storage area corresponding to the products to be picked, the location of the aisles within the storage area, and storage locations within the aisles. Step S102: generate at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch. Step S103: optimize the picking path of the picking task according to the picking strategy.

Herein, the processor 1001 comprises one or more processors. Each of these processors can be an integrated circuit chip with the ability to process signals. The above processor 1001 can be a general-purpose processor, including a Central Processing Unit (CPU), a Micro Controller Unit (MCU), a Network Processor (NP), or other conventional processors; it can also be a special-purpose processor, including a Neural-network Processing Unit (NPU), a Graphics Processing Unit (GPU), a Digital Signal Processor (DSP), an Application Specific Integrated Circuit (ASIC), a Field Programmable Gate Array (FPGA), or other programmable logic devices, discrete gate or transistor logic devices, and discrete hardware components. Moreover, when there are multiple processors 1001, some of them can be general-purpose processors, and the others can be special-purpose processors.

The memory 1003 comprises one or more memories. Each of these memories can be, but is not limited to, Random Access Memory (RAM), Read Only Memory (ROM), Programmable Read-Only Memory (PROM), Erasable Programmable Read-Only Memory (EPROM), Electrically Erasable Programmable Read-Only Memory (EEPROM), etc.

It can be understood that the structure shown in Figure 10 is only schematic. The electronic device 1000 may also comprise more or fewer components than those shown in Figure 10, or have a configuration different from that shown in Figure 10. The components shown in Figure 10 can be implemented by means of hardware, software, or a combination thereof. In the embodiments of the present application, the electronic device 1000 can be, but is not limited to, physical devices such as a desktop computer, a laptop computer, a smartphone, a smart wearable device, and a vehicle-mounted device, and can also be virtual devices such as a virtual machine. In addition, the electronic device 1000 is not necessarily a single device, and can also be a combination of multiple devices, such as a server cluster, and so on.

The embodiment of the present application also provides a computer-readable storage medium. The computer-readable storage medium stores computer program instructions. When the computer program instructions are run by a computer, the computer will execute the method for optimizing the picking path described in the foregoing method embodiments.

In the embodiments provided in the present application, it should be understood that the disclosed device and method can be implemented in other ways. The device embodiments described above are merely illustrative. For example, the division of units is only a logical functional division, and there may be other division methods in actual implementation. For another example, multiple units or components can be

combined or integrated into another system, or some features can be ignored or not executed. In addition, the coupling, direct coupling, or communication connection shown or discussed among them can be an indirect coupling or communication connection through some communication interfaces, devices, or units, and it can be in the form of electrical, mechanical, or other types.

In addition, the units described as separate components may or may not be physically separated, and the components displayed as units may or may not be physical units, that is, they may be located in one place, or may be distributed over multiple network units. Some or all of the units can be selected according to actual needs to achieve the purpose of the solution of this embodiment.

Furthermore, the functional modules in the various embodiments of the present application can be integrated together to form an independent part, or each module can exist separately, or two or more modules can be integrated to form an independent part.

It should be noted that if the functions are implemented in the form of software functional modules and sold or used as independent products, they can be stored in a computer-readable storage medium. Based on this understanding, the technical solution of the present application, in essence, or the part that contributes to the existing technology, or part of the technical solution can be embodied in the form of a software product. The computer software product is stored in a storage medium, and comprises several instructions for enabling a computer device (which can be a personal computer, a server, a network device, etc.) to execute all or part of the steps of the methods described in the various embodiments of the present application. The foregoing storage media comprise various media that can store program code, such as a USB flash drive, a mobile hard disk, a Read-Only Memory (ROM), a Random Access Memory (RAM), a magnetic disk, or an optical disk.

In this text, relational terms such as the first and the second are only used to distinguish one entity or operation from another entity or operation, and do not necessarily require or imply that there is any such actual relationship or order between these entities or operations.

The above are only the embodiments of the present application, and are not intended to limit the protection scope of the present application. For those skilled in the art, the present application can have various modifications and changes. Any modifications, equivalent replacements, improvements, etc. made within the spirit and principles of the present application should be included in the protection scope of the present application.

What is claimed is:

1. A method for optimizing a picking path, comprising steps of:

obtaining a storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises a location of a storage area corresponding to the products to be picked, locations of aisles within the storage area, and storage locations within the aisles;

generating at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch; and

optimizing the picking path of the picking task according to a picking strategy.

2. The method for optimizing a picking path according to claim 1, wherein the step of optimizing the picking path of the picking task according to a picking strategy comprises:

optimizing the in-aisle path of the picking task according to the picking strategy to obtain an in-aisle picking sequence within the aisle; and

optimizing an inter-aisle path of the picking task according to the in-aisle picking sequence and a shortest path rule to obtain an in-storage area picking sequence within the storage area.

3. The method for optimizing a picking path according to claim 2, wherein the picking strategy comprises at least one of the following:

selecting the picking path with the shortest distance within the aisle;

selecting the picking path with the fewest times of climbing ladders within the aisle;

selecting the picking path with the fewest times of squatting within the aisle; and

selecting the picking path with the fewest times of turning around within the aisle.

4. The method for optimizing a picking path according to claim 2, wherein the step of optimizing an inter-aisle path of the picking task according to the in-aisle picking sequence and a shortest path rule to obtain an in-storage area picking sequence within the storage area comprises steps of:

constructing a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule; and

determining the in-storage area picking sequence by solving the source-free shortest path problem.

5. The method for optimizing a picking path according to claim 4, wherein the step of constructing a source-free shortest path problem according to the in-aisle picking sequence and the shortest path rule comprises:

determining two nodes corresponding to the aisle according to the in-aisle picking sequence corresponding to the aisle, wherein the nodes represent the shortest path

within the aisle covering the storage locations corresponding to the in-aisle picking sequence, and a picking sequence corresponding to the two nodes is opposite; and

the step of determining the in-storage area picking sequence by solving the source-free shortest path problem comprises:

calculating an adjacent distance between the nodes of two adjacent aisles;

determining multiple target nodes that minimize a distance from the first aisle to the last aisle according to the adjacent distance, wherein each aisle corresponds to one target node; and

determining the in-storage area picking sequence according to the multiple target nodes.

6. The method for optimizing a picking path according to claim 1, wherein the step of generating multiple picking tasks according to the storage location layout information comprises a step of:

for an order batch, using a greedy algorithm to generate the multiple picking tasks according to the storage location layout information.

7. The method for optimizing a picking path according to claim 6, wherein the step of using a greedy algorithm to generate the multiple picking tasks according to the storage location layout information comprises:

determining a seed product, wherein the seed product is the product to be picked that has the most common aisles with other products to be picked among the products to be picked that have not been arranged, and has the smallest number of aisles, columns, and layers;

initializing a current picking task with the seed product;

determining an insertion product from the products to be picked that have not been arranged using a greedy rule, wherein the insertion product is the product to be picked that has the lowest insertion cost when inserted into the current picking task; and

inserting the insertion product into the current picking task until number of the products to be picked in the current picking task is greater than a number threshold.

8. The method for optimizing a picking path according to claim 7, wherein the insertion cost represents an increase in the picking path after inserting the product to be picked.

9. The method for optimizing a picking path according to any one of claims 1-8, wherein the step of generating at least one order batch according to the storage location layout information comprises steps of:

determining whether the order to be planned comprises multiple products to be picked; and

if the order to be planned comprises multiple products to be picked, dividing the multiple orders to be picked into multiple order batches according to the storage

location layout information and batching conditions; otherwise, generating one order batch according to all the orders to be planned.

10. The method for optimizing a picking path according to claim 9, wherein the step of dividing the multiple orders to be picked into multiple order batches according to the storage location layout information and batching conditions comprises steps of:

determining the order batch that covers the smallest number of the storage areas according to constraint conditions,

wherein the constraint conditions comprise: one order to be picked is assigned to one order batch; number of the orders to be picked in one order batch is not greater than the maximum number of the orders to be picked that can be assigned to the order batch; number of the products to be picked for one storage area in one order batch is not greater than a total number of the orders to be picked.

11. The method for optimizing a picking path according to claim 9, wherein the step of dividing the multiple orders to be picked into multiple order batches according to the storage location layout information and batching conditions comprises:

constructing a product location set corresponding to the orders to be picked, wherein values in the product location set represent storage area numbers where the products to be picked in the orders to be picked are located;

sorting the values in the product location set and concatenating them into a string;

sorting the multiple orders to be picked according to the strings corresponding to the multiple orders to be picked; and

dividing the sorted orders to be picked according to the maximum number of the orders to be picked that can be assigned to the order batch.

12. An apparatus for optimizing a picking path, comprising:

an acquisition module, configured to obtain a storage location layout information corresponding to multiple orders to be planned, wherein each order to be planned comprises one or more products to be picked, and the storage location layout information comprises a location of a storage area corresponding to the products to be picked, locations of aisles within the storage area, and storage locations within the aisles;

a generation module, configured to generate at least one order batch and a picking task corresponding to each order batch according to the storage location layout information, wherein each order to be planned is included in one order batch; and

an optimization module, configured to optimize the picking path of the picking task according to a picking strategy.

13. An electronic device, comprising: a processor, a memory, and a bus; wherein the processor and the memory communicate with each other through the bus;

the memory stores computer program instructions executable by the processor, and the processor is capable of executing the method for optimizing a picking path according to any one of claims 1-11 by invoking the computer program instructions.

14. A computer-readable storage medium, storing computer program instructions, wherein the computer program instructions, when run by a computer, enables the computer to execute the method for optimizing a picking path according to any one of claims 1-11.