参赛队员姓名：王雨舟

中学：山东省实验中学

省份：山东

国家/地区：中国

指导教师姓名：马建华，石磊

论文题目：Path Optimization of Takeaway Distribution Based on Artificial Bee Colony Algorithm

Path Optimization of Takeaway Distribution Based on Artificial Bee Colony Algorithm

Wang Yuzhou

Abstract

In this paper, we first introduce the formulation of the path optimization problem, then we describes an optimization takeaway delivery path model. This model adopts split algorithm and bipartite graph matching algorithm to calculate the best delivery scheme, makes use of Artificial Bee Colony algorithm to get the best permutation with time and capacity constraints to acquire the greatest satisfaction with least human resources, and minimizes the maximum delay time and relatively balanced number of orders for each deliveryman. Also, in this paper, we give a real-life example to test and verify the feasibility of the model.

Keywords: artificial bee colony algorithm; takeaway food distribution; optimization model; path optimization; e-commerce terminal logistics distribution;

Contents

1 Introduction

With the ever-accelerating progress of the network technique, the food delivery industry has gained much popularity. An increasing number of people start to order delivery food via certain platforms. Currently, the main method of order delivery food is that consumers connect the restaurant straightforward on the platform, at the same time, the platform distributes the missions to the deliverers. However, the deliverer cannot proceed the missions properly due to the particularity of this implement. It's always a critical aporia that how to properly distribute the missions to the deliverers since effective and efficient delivery is not only the foundation of e-business delivery but the crux of improving the quality of delivery. When offering delivery service, the platform will stipulate the time limit to ensure the arrival on time and the performance of the deliverers. The platform usually implements the worst option - squandering human resources to deliver the food, which harms the fierce competition. As a result, setting time and the maximum number of orders that one deliverer can distribute as the constraint condition to cut down the used number of deliverers as low as possible has a huge impact on the platform, achieving energy-efficient missions for it.

The earliest problem mentioned path optimization of takeaway distribution is the Vehicle Routing Problem, which can be recognized as the simplest version of this problem. It can be described as designing the least weight delivery routes through a set of geographically scattered customers or positions with several side restrains. Decades have passed since the publication of this paradox in 1959 by Dantzig and Ramser, who simultaneously delivered a simple, inexact, matching-based method for the solution. Various implements have been designed in the following years. The most famous heuristic may be Clarke and Wright's savings heuristic, considering as the most leading algorithm because of its speed. More specifically, the initial exact algorithm was proposed in 1981 by Christofides and several years later, a method based on linear programming was proposed[1]. Entering the 21st century, a variety of exact algorithms based on mathematical interpretation have been announced. Meanwhile, derivatives of the standard VRP problems was proposed along with the development of the algorithm, such as capacitated vehicle routing problem(CVRP), vehicle routing problem with time windows(VRPTW), multi-depot vehicle routing problem(MDVRP), etc.

VRP problems in the nineties flourished because of the emergence of modern heuristic approaches. Also, it has promoted several algorithms' understanding, growth, and development. Taboo search, ant colony algorithm, and artificial bee colony algorithm are applied in VRP. The best way to search is by wide and deep search in the solution domain. As a result, researchers from worldwide applied and combined different algorithms. Berthold, Ochi utilized genetic algorithm, M.Lang and S.Hu derived combined genetic algorithm. However, papers in the domain of takeaway food delivery are still scarce. These years in China scholars published several papers along with the development of the Chinese delivery food distribution industry. W.Gao adopted a firework algorithm to select routes of delivery with consideration of safety[3]. J.Zhai and Y. Tai exploited genetic algorithms and finished the problem including time window constraints and customers' special needs[4].

The artificial bee colony algorithm discussed in this paper is a swarm-based meta-heuristic algorithm introduced by D.Karaboga in 2005, which is based on the foraging behavior of bees[5]. This algorithm is first implemented on function numerical optimization. Due to its excellent performance, comparable with other meta-heuristic algorithms, and rather a simple process, it is capitalized on various numerical optimization problems, including neural network training[6], image processing[7], etc. Also, it can be employed on discrete optimization, involving standard VRP [8], minimum spanning tree[9], etc. It is worth to mention that A.Banharnsakun proposed the solution to the TSP problem, where the author extended the neighborhood search from a regular method to crossing route based on greedy algorithm. Meanwhile, several scholars proposed improvement aiming at the algorithm itself. Y. Xue proposed global optimized artificial bee colony algorithm[18] and D.Karaboga proposed combinational artificial bee colony algorithm[17].

However, consider the real-life experience, according to the actual observation, most companies do not consider to cut the human resource budget and do not choose to employ as many deliverers as possible because of severe constraints that if a customer's order arrives late, the company will compensate money and potential customer churn. Though it does pledge that orders are distributed on time, companies dissipate unnecessary costs. Under some special circumstances, the maximum delay is too high to tolerate for customers, causing additional customer churn. If platforms can aiming to reduce human resources and maximum delay time, reaching a relatively balanced status for orders, the performance and competitiveness will be greatly enhanced for the takeaway distribution industry.

This paper investigates artificial bee colony algorithm as a searching method, split algorithm to reduce manpower as low as possible and bipartite graph matching to curtail the maximum delay time to achieve balanced distributed orders. the optimization takeaway delivery path model for the food delivery industry is developed where the artificial bee colony algorithm is applied to determine the sequence of the delivery and is combined with other implements to solve the optimization problem. A real application of Wanda Plaza in Jingqi Road, Jinan is modeled and analyzed and the optimized result is obtained, the model and algorithm developed in this paper can be used in food delivery companies.

2 Question Analysis and Modeling

Consider a certain food court as the only depot, there are m deliverers serve for the platform. The average speed of the deliverers is v and the maximum capacity is G.

Customers make orders via the platform and the platform distributes missions by section to deliverers. Assume that the set of orders placed in a time period is S, each order has a capacity of $c_i, i \in S$. The distance between customer i and j is $d_{ij}, i, j \in S$. Customers distance between food court is $d_{0i}, i \in S$.

Some customers have arrival time limits. If the customer does not set a limit, the platform commits that each order should be distributed in a certain amount of time. Each customer has a required time of arrival, denoted by $t_i^1, i \in S$.

When arranging missions, all the deliverers are working, the mission have been distributed is denoted by $a_k, k = 1, 2, 3, \ldots, m$. The start time, which the deliverers' time of return to the food court, is denoted by $t_k^2, k = 1, 2, \ldots, m$.

When distributing missions, the first element into consideration is the time limit. If mission occupies a long time, the lower the delay time, the better. If the time does not surpass the time limit, then the delay time is 0.

Given a distribution scheme, each customer's delay time is whether 0 or a positive number and the maximum value of the delay time is the maximum delay time. The objective is to minimize the maximum delay time.

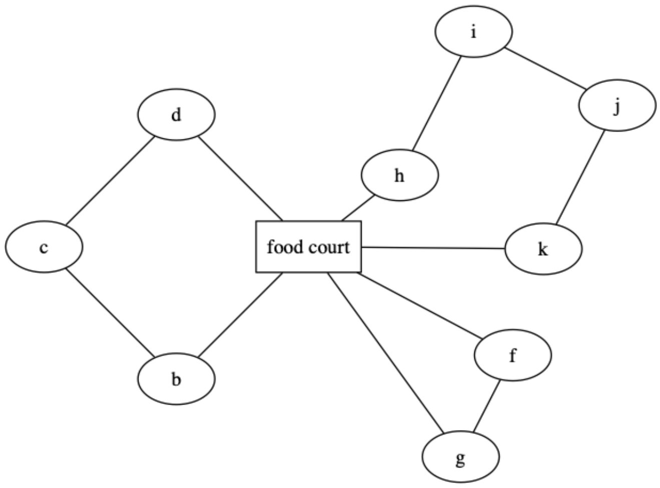

Figure 1: Example of distribution problem

The income of each deliverer is proportional to his or her orders, and as a result, the balanced mission should be taken into consideration to bridge the gap between income inequality.

In conclusion, this problem will select deliverers, distribute the orders and the sequence of orders.

2.1 Analysis

Let k-th deliverer's availability be $x_k, k = 1, 2, \ldots, m$. The indicator that k-th deliverer chooses customer i is denoted by $y_{ki}, k = 1, 2, \ldots, m, i \in S$, where 1 denotes chosen and 0 denotes not chosen. If do not choose deliverer k, then the capacity is 0, otherwise, it cannot exceed G.

$$\sum_{i \in S} c_i y_{ki} \leq G x_k, k = 1, 2, \ldots, m \tag{2.3}$$

A customer can only be distributed by only one deliverer.

$$\sum_{k=1}^{m} y_{ki} = 1, i \in S \tag{2.4}$$

Each deliverer should plan the route of delivery, that is, plan the sequence of customers. As for the $k-th$ deliverer who distributes customers i and j, set variable $z_{ij}^k, k = 1, 2, \ldots, m, i, j \in S$. If the route is from i to j, then $z_{ij}^k = 1$, otherwise is 0. Only when i and j are served by k can they can be on the same route, as a result:

$$z_{ij}^k \leq (y_{ki} + y_{kj})/2, k = 1, 2, \ldots, m, i, j \in S \tag{2.5}$$

$$z_{0i}^k \leq y_{ki}, k = 1, 2, \ldots, m, i \in S \tag{2.6}$$

$$z_{i0}^k \leq y_{ki}, k = 1, 2, \ldots, m, i \in S \tag{2.7}$$

Each customer can only enter and exit once:

$$\sum_{k=1}^{m} \left(\sum_{j \in S} z_{ij}^k + z_{i0}^k \right) = 1, i \in S \tag{2.8}$$

$$\sum_{k=1}^{m}\left(\sum_{j\in S} z_{ji}^{k}+z_{0i}^{k}\right)=1, i\in S \tag{2.9}$$

To avoid one deliverer's route emerges two circuits, the route should satisfy:

$$u_i-u_j\geq 1+R\left(z_{ij}^{k}-1\right), k=1,2,\ldots,m, i\in S, j\in S\cup\{0\} \tag{2.10}$$

in which R is a big positive number and u_i is the potential of i.

Set the arrival time of i to be $t_i, i\in S$, the arrival time of the food court is $t_0=0$, so:

$$t_i=\sum_{\substack{j\in S\\ j\neq i}}\left(\left(t_j+\frac{d_{ji}}{v}\right)\sum_{k=1}^{m} z_{ji}^{k}\right)+\sum_{k=1}^{m}\left(\frac{d_{0j}}{v}+t_k^2\right) z_{0i}^{k}, i\in S \tag{2.11}$$

Set the delay time for $i-th$ customer to be $t_i^3, i\in S$, so:

$$t_i^3=\max\left(0, t_i-t_i^1\right), i\in S \tag{2.12}$$

The number of customers that the deliverer will distribute is :

$$a_k+\sum_{i\in S} y_{ki}, k=1,2,\ldots,m \tag{2.13}$$

So the objective functions are:

$$\min\max_{i\in S} t_i^3 \tag{2.1}$$

$$\min\left(\max\{a_k+\sum_{i\in S} y_{ki}\|k=1,2,\ldots,m\}-\min\{a_k+\sum_{i\in S} y_{ki}\|k=1,2,\ldots,m\}\right) \tag{2.2}$$

2.2 Modeling

$$\min \quad \max_{i \in S} t_i^3 \tag{2.1}$$

$$\min \quad \left(\max\{a_k + \sum_{i \in S} y_{ki} \| k = 1, 2, \ldots, m\} - \min\{a_k + \sum_{i \in S} y_{ki} \| k = 1, 2, \ldots, m\} \right) \tag{2.2}$$

$$s.t. \quad \sum_{i \in S} c_i y_{ki} \leq G x_k, k = 1, 2, \ldots, m \tag{2.3}$$

$$\sum_{k=1}^{m} y_{ki} = 1, i \in S \tag{2.4}$$

$$z_{ij}^k \leq (y_{ki} + y_{kj})/2, k = 1, 2, \ldots, m, i, j \in S \tag{2.5}$$

$$z_{0i}^k \leq y_{ki}, k = 1, 2, \ldots, m, i \in S \tag{2.6}$$

$$z_{i0}^k \leq y_{ki}, k = 1, 2, \ldots, m, i \in S \tag{2.7}$$

$$\sum_{k=1}^{m} \left(\sum_{j \in S} z_{ij}^k + z_{i0}^k \right) = 1, i \in S \tag{2.8}$$

$$\sum_{k=1}^{m} \left(\sum_{j \in S} z_{ji}^k + z_{0i}^k \right) = 1, i \in S \tag{2.9}$$

$$u_i - u_j \geq 1 + R\left(z_{ij}^k - 1\right), k = 1, 2, \ldots, m, i \in S, j \in S \cup \{0\} \tag{2.10}$$

$$t_i = \sum_{\substack{j \in S \\ j \neq i}} \left(\left(t_j + \frac{d_{ji}}{v} \right) \sum_{k=1}^{m} z_{ji}^k \right) + \sum_{k=1}^{m} \left(\frac{d_{0j}}{v} + t_k^2 \right) z_{0i}^k, i \in S \tag{2.11}$$

$$t_i^3 = \max\left(0, t_i - t_i^1\right), i \in S \tag{2.12}$$

$$a_k + \sum_{i \in S} y_{ki}, k = 1, 2, \ldots, m \tag{2.13}$$

This model is a non-linear integer programming that has a large number of variables and constraints. Utilizing regular approaches will not gain a feasible result. Instead, this paper will consider evolutionary algorithms.

3 Algorithm Analysis

This problem involves divisions of customers, distribution of mission and the sequence of delivery. How to determine the sequence is the vital key to this problem. However, when given a sequence, the division of customers and the distribution of mission is relatively simple.

As a result, this paper utilizes artificial bee colony algorithm to determine the sequence of the delivery and uses other implements to solve the objective function.

These two objectives have different priorities. As a result, in determining the value of the objective function, each function value is weighted by a certain coefficient.

The main algorithm of this paper is artificial bee colony algorithm(ABC), which is composed of three distinct parts: employed bees, unemployed bees and food sources. In a sentence, this algorithm can be described as the process of approaching the best food source.

There are three flocks of bees in the artificial bee colony algorithm: employed bees search in a specific food source, onlooker bees which observe the dance of employed bees and search in the neighborhood and scout bees which randomly search for food sources when the food source is depleted. In the beginning, the initial food source is found by scout bees. Then employed bees receive the signal of scout bees and start to search around the food sources but ceaselessly collecting honey will drain the food source. At that time, employed bees will transform into scout bees to search another food source. In the algorithm, the position of a food source represents a feasible solution, the amount of honey represents the quality of the food source and the number of employed bees represents the number of solutions.

3.1 Key Techniques

3.1.1 Coding Method

A feasible solution vector is denoted as $\vec{x}_{ij} = (x_1, x_2, \ldots, x_N)$,, where $\vec{x}_{ij}$ represent the $i-th$ generation's $j-th$ food source, and $x_1, x_2, \ldots, x_N$ denotes the full permutation of all customers.

First define function $random\_shuffle(\vec{x})$:

Algorithm 1 $random\_shuffle(\vec{x})$

```
count ⇐ size of x⃗
for i ⇐ 0 to count do
    swap(x[i], x[random(i, count)])
end for
```

where $swap(a, b)$ denotes that values of a and b are swapped and $random(i, j)$ denotes that generate a random number between i and j.

Assume a solution vector is $\vec{x} = (a_1, a_2, \ldots, a_N)$, according to constraints and parameters, all employed bees transform into scout bees, thus randomly search for the vector $\vec{x}_m(m = 1...SN$, SN :)the size of the population. The initial solution vector is determined by this formula:

$$\vec{x} = random\_shuffle(1, 2, \ldots, N) \tag{3.1}$$

3.1.2 Calculating Objective Function

This step is divided into two parts. First, cut the route by the split algorithm and choose the deliverers according to the start time.

I. Use split algorithm to determine the shortest path with the least edges.

Generate foundation graph with capacity constraint, and then calculate each customer's arrival and delay time, and set the maximum delay time as the weight of the edge. Then delete the edges from large to small until there is no route whose number of edges is less than or equal to m. Then the former step's result is the best solution to this problem. Steps are:

First step: generate the foundation graph with capacity constraints, denoted as $w = \infty, ow = \emptyset$

Second step: calculate each customer's arrival time and delay time.

Third step: calculate each edge's maximum delay time as the weight of the edge.

Fourth step: determine the route that as least number of edges, if there is no route of the number of edges is greater than m, then stop calculating and print w and ow, otherwise, turn to the next step.

Fifth step: find the maximum value of weight among edges of the route, let w equals the maximum value of the route and ow record the best solution. Delete all the edges greater than or equal to w, turn to step four.

II. Generate a bipartite graph, the vertices represent deliverers and routes, the weight is the sum of delayed time and the start time of the deliverers.

Delete the edges from large to small after calculating the maximum matching until the pair of matching is less than m.

First step: generate the bipartite graph, denote $p = \infty, op = \emptyset$.

Second step: calculate the weights of each edge which equal the sum of delayed time and the start time of the deliverers.

Third step: determine the maximum matching of the bipartite matching of the graph. If the matching number is less than m, stop the algorithm and print p, op. Otherwise, turn to the next step.

Fourth step: search the maximum weight of the matching and set p equal to the maximum weight of the matching, op is the best solution. Delete all the edges greater or equal to p and turn to step 3.

That calculate the number of customers that each deliverer will serve and determine the difference between the largest and the least number of customers is the objective function.

Example:

Assume a feasible solution is $\vec{x}_1 = (1, 4, 6, 5, 2, 3)$, the capacities are $\vec{c} = (30, 20, 60, 10, 50, 20)$, $G = 100$, $v = 50$ and the time limit is 10, the following foundation graph can be generated.

i	1	2	3	4	5	6
0	300	600	700	500	300	650

Table 1: Values of d_{0i}

d_{14}	d_{46}	d_{65}	d_{52}	d_{23}
400	600	900	300	100

Table 2: Values of d_{ij}

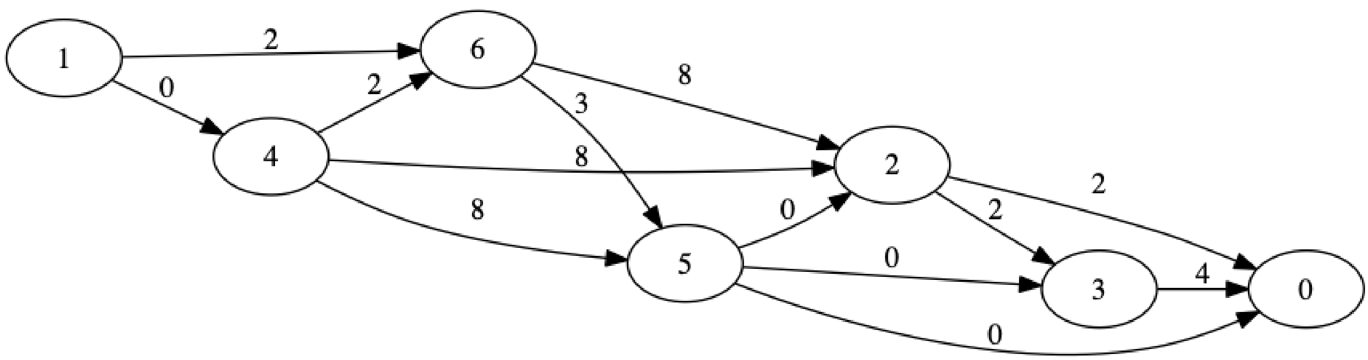

Figure 2: foundation graph

Then, use the heap optimized Dijkstra algorithm, determine the route from point 1 to point 0 which has least number of edges. In this example, the route is $1->4->5->0$. Find the largest edge in this path and delete all the edges greater than or equal to this edge. Which is transform to the following graph:

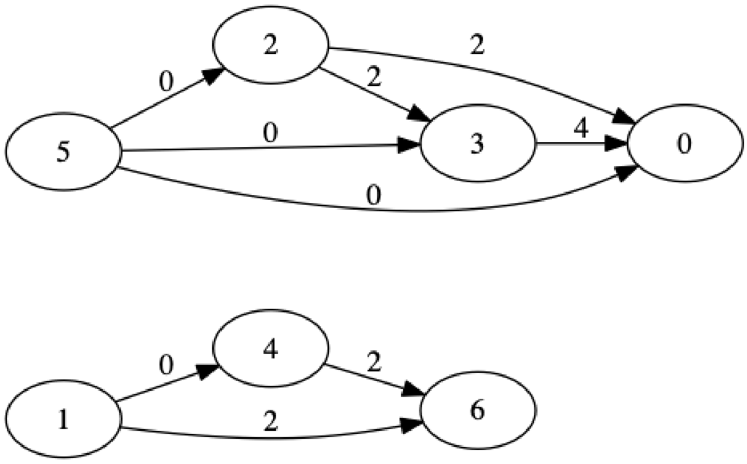

Figure 3: Graph After First Iteration in SPILT

However, this graph becomes an unconnected graph, indicating that the route in former step is the best route.

The time complexity is $O(n^2 \log n)$.

Assume there are four deliverers in this area, the starting time of each deliverer is $(2, 5, 4, 7)$. The points denoted with a are the deliverer and b are the routes. The first step is to connect all the points from a to b.

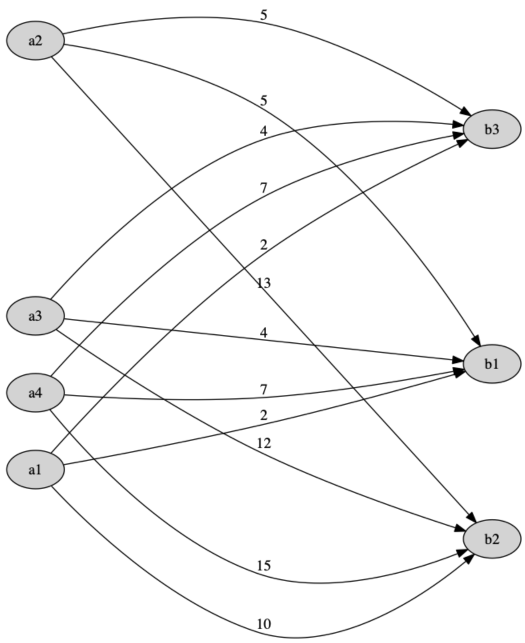

Figure 4: foundation bipartite graph

Randomly match the bipartite graph, assume the results are $a1-b3, a3-b2, a4-b1$, where the largest edge possesses weight of 12. Delete all the edges greater than or equal to this edge. The result is:

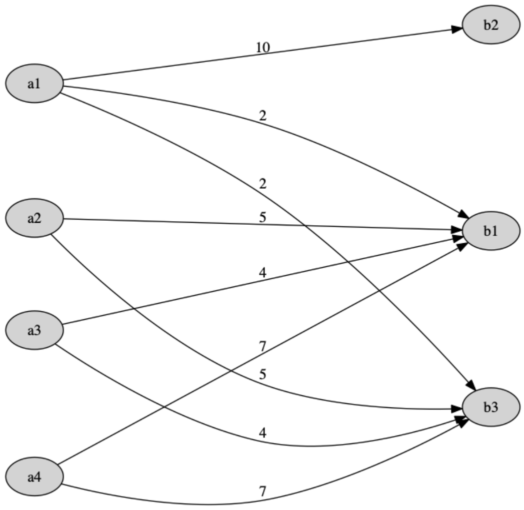

Figure 5: Graph after first iteration in the matching

Randomly match again, assume the results are $a1 - b2, a2 - b3, a4 - b1$, where the largest edge possesses weight of 10. Delete all the edges greater than or equal to this edge. The result is:

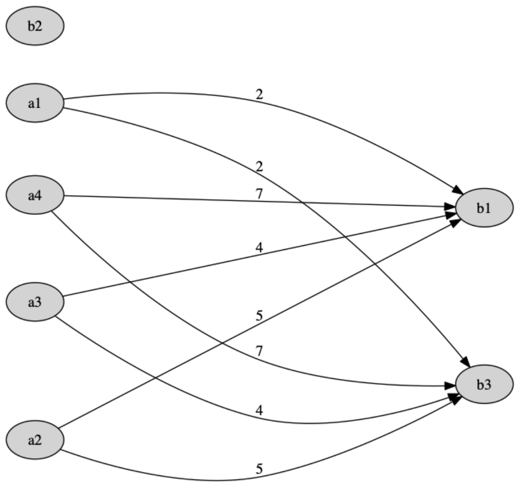

Figure 6: Graph after second iteration in the matching

where point $b2$ cannot be matched. Consequently, the result in former step is the best solution. That is, the best solution is $a1 - b2, a2 - b3, a4 - b1$.

The time complexity is $O(n^4)$ and the total time complexity is $O(n^4)$.

The value of the objective function is $(3 - 1) * 0.5 + 10 = 11$.

3.1.3 Selection Method of Onlooker Bees

Employed bees transmit the information of the food source, the quality of the food source, by dancing to the onlooker bees, who select the food source by chance. In the algorithm, this step is usually performed by Roulette method.

The possibility, p_m, of select food source, $\vec{x}_m$, is determined by the following formula.

$$p_m = \frac{fit_m(\vec{x}_m)}{\sum\limits_{m=1}^{SN} fit_m(\vec{x}_m)} \tag{3.2}$$

where $fit_m(\vec{x}_m)$ is the function value of $\vec{x}_m$, SN is the population size.

3.1.4 Neighborhood Search

The discretization of neighborhood search is based on the concept of swap operator and swap sequence based on K.Wang's research[10]. The defined swap operator is the alternative of traditional neighborhood search. Define swap operator $SO(i_1, i_2)$ denotes that to swap the position of point a_{i1} and a_{i2} in the solution vector. As a result, $\vec{x}_{new} = \vec{x}+SO(i_1, i_2)$, whose $+$ is not the original meaning. For example, assume the solution vector is $\vec{x}_1 = (2, 6, 4, 3, 1, 5, 7)$, swap operator is $SO(2, 6)$, thus the new solution is $\vec{x}_{new} = \vec{x}_1 + SO(2, 6) = (2, 5, 4, 3, 1, 6, 7)$.

Define swap sequence SS as a queue which composes of multiple swap operators, denoted as $SS = (SO_1, SO_2, \dots, SO_n)$. In this problem, the sequence will continuously update so it can be described as $\vec{x}_{new} = \vec{x} + SS = (((\vec{x} + SO_1) + SO_2) + \cdots + SO_N)$.

3.1.5 Scout Method

When a certain employed bee's food source's quality cannot progress after a specific iteration, this food source will be abandoned and the employed bee will transform into scout bee and randomly search for food source according to (3.1).

3.2 Corresponding Relation

Definition of Artificial Bee Colony Algorithm	Definition in this problem
food source	solution vector
quality of food source	objective function
neighborhood search	swap operator
best food source	the solution with least function value

Table 3: Corresponding Relation

3.3 Algorithm Steps

Step 1: Set the parameters, including bee colony size $size$, maximum iteration times $time$, replacement possibility p, maximum replacement time $limit$,current replacement time $still_i$ and current iteration time $k = 0$.

Step 2: Randomly generate $m/2$ permutation $\vec{x}_i$ as the food source of the employed bees, use the improved split algorithm and the maximum matching algorithm to calculate each food source's objective function $f_i, i = 1, 2, \dots, \frac{m}{2}$. Record the maximum value of the objective function, denoted as of, and the corresponding permutation is denoted as ox.

Step 3: Use Roulette method (formula (3.2)) according to the objective functions value of permutations to determine the number of onlooker bees k_i.

Step 4: Every employed bee and its onlooker bees implement neighborhood search and randomly generate swap operator SO_i. The new solution vector is $\vec{x}_{i^1} = \vec{x}_i + SO_1$. Determine the solution which has least function value $fmin_i = \min f_{ij}$ and record the corresponding permutation ox_i. If $fmin_i < f_i$, use the permutation ox^i to replace the food source $\vec{x}_i$, $f_i = fmin_i, still_i = 0$. Otherwise, generate random number r. If $r < p$, then replace the former permutation, $still_i = 0$, or

$still_i = still_i + 1, i = 1, 2, \ldots, \frac{m}{2}$.

Step 5: For every onlooker bee, if $still_i \geq limit$, then randomly generate a permutation $\vec{x}_i$ to replace the former food source and calculate the objective function f_i.

Step 6: $k = k + 1$. Determine the minimum value of each food source ofk and record the corresponding permutation oxk. If $ofk < of$, then $of = ofk, ox = oxk$.

Step 7: If $k \leq T$, turn back to step 3, otherwise, stop the algorithm steps and print the results ox and of.

3.4 Algorithm Steps Pseudocode

The below algorithms shares global variables.

Algorithm 2 $Dijkstra(g, s, rec)$

```
set que ⇐ a Min-heap with first element value and second element number, where que.push(a, b)
denotes pushing a element with value a and number b
∀i, d_i ⇐ ∞, vis_i = 0
d_s = 0, que.push(0, s)
while que is not empty do
    x ⇐ the root number of que
    pop the root
    if vis_x = 1 then
        go to next circulation
    end if
    vis_x ⇐ 1
    for i ⇐ 1 to the number of connected nodes of x do
        v ⇐the i-th connected node of x
        if d_v > d_x + 1 then
            d_v ⇐ d_x + 1, que.push(d_v, v)
        end if
    end for
end while
```

Algorithm 3 $bibartite\_graph\_matching(g)$

```
set global mch_i ⇐the match result, dfn_i ⇐the timestamp, ans ⇐the number of matchings.
for i ⇐ 1 to the number of nodes in the g do
    if dfs(i, i) =true then
        ans = ans + 1
    end if
end for
```

Algorithm 4 $dfs(u, vis)$

for $i \Leftarrow$ the i-th connected point of u **do**
  if $dfn_i \neq vis$ **then**
    $dfn_i \Leftarrow vis$
    if $mch_i = 0$ or $dfs(mch_i, vis)$ **then**
      $mch_i \Leftarrow u$
      return true
    end if
  end if
  return false
end for

Algorithm 5 $calculate\_fitness(\vec{x})$

generate temporary fundamental graph gr according to capacity
while the route between start point and point 0 exists **do**
  set a temporary graph gr_t, which is identical as gr but weights are all 1.
  $Dijkstra(gr_t,$ start point$)$
  set $max\_weight \Leftarrow \max_{edge \in route} weight$
  delete all the edges $\leq max\_weight$
end while
Generate a bipartite graph g where the left side denotes the deliverers and the right side denotes orders. All the nodes from left are connected with all the nodes from right. The weight are the time of the deliverer plus the delayed time of the orders.
while the maximum number of matching equal to the split route number **do**
  $bibartite\_graph\_matching(g)$
  set $max\_weight \Leftarrow \max_{edge \in mch_{\text{left side of the graph}}} weight$
  delete all the edges $\leq max\_weight$
end while
return objective function value

Algorithm 6 $neighborhood\_search(i, j, p)$

```
fit1_ij ⇐ 1/calculate_fitness(x⃗_ij)
generate SO(i_1, i_2)
x⃗_t ⇐ x⃗_ij + SO(i_1, i_2)
fit2 ⇐ 1/calculate_fitness(x⃗_t)
if fit1 < fit2 then
   x⃗_ij ⇐ x⃗_t , still_j ⇐ 0
else
   generate a random number r
   if r < p then
      x⃗_ij ⇐ x⃗_t , still_j ⇐ 0
   end if
else
   still_j ⇐ still_j + 1
end if
```

Algorithm 7 $optimization\ steps$

```
set time, limit, S, size, still_i, p
for i ⇐ 1 to m/2 do
   set x⃗_1i ⇐ (1, 2, ..., card(S))
   random_shuffle(x⃗_1i)
   f_1i ⇐ calculate_fitness(x⃗_1i)
   of = min_{i∈1,2,...,m/2} f_1i, ox ⇐ the solution vector of of
end for
for i ⇐ 1 to time do
   for j ⇐ 1 to size/2 do
      select food source by formula (3.2)
      neighborhood_search(i, j, p)
      if still_i = limit then
         random_shuffle(x⃗_ij)
      end if
      set ofk ⇐ min_{j∈1,2,...,m/2} f_ij, oxk ⇐ the solution vector of ofk
      if ofk < of then
         of ⇐ ofk, ox ⇐ oxk
      end if
   end for
end for
```

4 Example

Assume Wanda Plaza in Jingsi road, Jinan (denoted by red arrow) is the food court. The neighborhood area($2km^2$) has orders wait to be delivered.

Figure 2: Example

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
800	750	650	150	100	150	450	700	500	600	300	900	500	600	1000

Table 4: Value of d_{0i}

1	2	3	4	5	6	7
5	3	10	7	1	2	3

Table 5: Value of deliverers' time delay

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
50	100	70	10	80	50	80	100	120	60	30	30	80	100	60

Table 6: Value of c_i

(omit the table of d_{ij})

There are 7 deliverers who have capacity of 300, velocity of 100, an extra 4 minute to deliver an order and 8 minute of limit between orders in this area.

After iterations, the best solution vector is $\vec{x} = (9, 10, 3, 8, 1, 2, 11, 5, 4, 6, 7, 13, 14, 15, 12)$ and the best deliverers are $2, 5, 6, 7$.

The allocation of orders in the solution vector is $9, 10, 3, |8, 1, 2, |11, 5, 4, 6, 7, |13, 14, 15, 12$ and the function value is $2 * 0.5 + 1 = 2$.

5 Conclusion

This paper establishes an optimization model of path routing of takeaway food distribution based on the artificial bee colony algorithm to determine the permutation of the delivery, the split algorithm to identify the sections of distribution and the maximum matching algorithm to calculate the best matching of deliverers and sections of distributions. In the real-life example, the result reveals that the number of deliverers can be greatly reduced, freeing up budget and personnel for platforms, which indicates that the efficiency and the effectiveness of delivery can be greatly enhanced.

References

[1] Laporte, G., Toth, P., & Vigo, D. (2013). Vehicle routing: Historical perspective and recent contributions. EURO Journal on Transportation and Logistics, 2(1-2), 1-4. doi:10.1007/s13676-13-0020-6

[2] Dantzig GB, & Ramser JH (1959) The truck dispatching problem. Manag Sci 6:80

[3] Gao, W., & Jiang, G. (2018). The research based on the path optimization problem of takeaway delivery. Information & Communications.

[4] Zhai, J., & Tai, Y. (2018). Delivery Routing Optimization Based on Time Window Constraint. Logistics Sci-Tech. doi:1002-3100 (2018) 03-0015-04

[5] Karaboga, Dervis. (2005). An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report - TR06. Technical Report, Erciyes University.

[6] Karaboga, D., & Akay, B. (2007). Artificial Bee Colony (ABC) Algorithm on Training Artificial Neural Networks. 2007 IEEE 15th Signal Processing and Communications Applications. doi:10.1109/siu.2007.4298679

[7] Cuevas, E. (2015). Artificial Bee Colony (ABC) algorithm and its use in digital image processing. Inteligencia Artificial, 18(55), 50. doi:10.4114/intartif.vol18iss55pp50-68

[8] Bhagade, A. S., & Puranik, P. V. (2012). Artificial Bee Colony (ABC) Algorithm for Vehicle Routing Optimization Problem. International Journal of Soft Computing and Engineering (IJSCE), 2(2), may 2012.

[9] Wang, Y. (2018). Improving Artificial Bee Colony and Particle Swarm Optimization to Solve TSP Problem. 2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS). doi:10.1109/icvris.2018.00051

[10] Wang, K., Huang, L., Zhou, C., & Pang, W. (2003). Particle swarm optimization for traveling salesman problem. Journal of Jilin University (Science Edition), 4, Oct. 2003, 477-480.

[11] Prins, C., Labadi, N., & Reghioui, M. (2008). Tour splitting algorithms for vehicle routing problems. International Journal of Production Research, 47(2), 507-535. doi:10.1080/00207540802426599

[12] Li, L.,Cheng, Y., & Niu, B.(2011). A Discrete Artificial Bee Colony Algorithm for TSP Problem.Chinese Journal of Management Science,Vol. 19.Special Issue October.2011.

[13] Christofides N, Mingozzi A, & Toth P (1981). State-space relaxation procedures for the computation of bounds to routing problems. Networks 11:145–164

[14] Sobti, S. & Singla, P. (2013). Solving Travelling Salesman Problem Using Artificial Bee Colony Based Approach. International Journal of Engineering Research & Technology. Vol. 2 Issue 6, June - 2013.

[15] Clarke, G. & Wright, J. (1962). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. The Roots of Logistics, 229-244. doi:10.1007/978-3-642-27922-5_18

[16] Xue, Y., Jiang, J., Zhao, B. et al. Soft Comput (2018) 22: 2935. https://doi.org/10.1007/s00500-017-2547-1

[17] Karaboga, D., Gorkemli, B. (2019) International Journal on Artificial Intelligence Tools 28:01. https://doi.org/10.1142/S0218213019500040

致谢

衷心感谢指导老师山东财经大学管理科学与工程学院博士生导师马建华教授，在研究整体过程中给予的指导，衷心感谢山东省实验中学石磊老师在参赛方面的指导，衷心感谢济南大学管理科学与工程系系主任常相全教授在论文选题初期给予的指导和建议。

本文最初选题基于生活实际，在学校点外卖时，常常会发现有大量的外卖员挤在学校门口，然而一个外卖员往往送完一两份订单便离开学校。然而同学们订外卖一般都会选择离学校附近的万达广场附近的餐馆，且要求送达时间差距不大，送多份订单是完全可行的。

带着这个问题，常相全教授初步引导我提出了问题，马建华教授在推导模型，开拓思路方面给予了指导与建议，山东省实验中学的石磊老师在整体方面提出了建议与改进。

在此再次向给予帮助的专家和老师们表示诚挚的感谢！

此页为学术诚信声明

本参赛团队声明所提交的论文是在指导老师指导下进行的研究工作和取得的研究成果。尽本团队所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果。若有不实之处，本人愿意承担一切相关责任。

参赛队员：王雨丹　　　　指导老师：马建华 MA Jianhua

2019年 9月 14日