Complex & Intelligent Systems (2022) 8:4939–4953
https://doi.org/10.1007/s40747-022-00719-4

ORIGINAL ARTICLE

Modeling stochastic service time for complex on-demand food delivery

Jie Zheng[1] · Ling Wang[1] · Shengyao Wang[2] · Jing-fang Chen[1] · Xing Wang[1] · Haining Duan[2] · Yile Liang[2] · Xuetao Ding[2]

Received: 6 October 2021 / Accepted: 5 March 2022 / Published online: 29 April 2022

Abstract

Uncertainty is everywhere in the food delivery process, which significantly influences decision-making for complex on-demand food delivery problems, affecting delivery efficiency and customer satisfaction. Especially, the service time is an indispensable part of the delivery process impacted by various uncertain factors. Due to the simplicity and high accuracy requirement, we model the uncertain service time as a Gaussian mixture model (GMM). In detail, we transform the distribution estimation problem into a clustering problem by determining the probability of each data belonging to each component (each cluster as well). A hybrid estimation of distribution algorithm is proposed to intelligently solve the clustering problem with the criterion to optimize quality and simplicity simultaneously. First, to optimize the simplicity, problem-specific encoding and decoding methods are designed. Second, to generate initial solutions with good clustering results, a Chinese restaurant process-based initialization mechanism is presented. Third, a weighted-learning mechanism is proposed to effectively guide the update of the probability model. Fourth, a local intensification based on maximum likelihood is used to exploit better solutions. The effect of critical parameters on the performances of the proposed algorithm is investigated by the Taguchi design of the experimental method. To demonstrate the effectiveness of the proposed algorithm, we carry out extensive offline experiments on real-world historical data. Besides, we employ the GMMs obtained by our algorithm in a real-world on-demand food delivery platform, Meituan, to assist decision-making for order dispatching. The results of rigorous online A/B tests verify the practical value of introducing the uncertainty model into the real-life application.

Keywords Complex on-demand food delivery · Stochastic service time modeling · Gaussian mixture model · Hybrid estimation of distribution algorithm

Introduction

Under the prosperous development tendency of e-commerce, on-demand food delivery (OFD) service has become a wave worldwide. In recent years, many OFD platforms have sprung up all over the world, such as Meituan (China), Grubhub (USA), Deliveroo (UK), and Swiggy (India). According to the Statista Digital Market Outlook, the global revenues of the OFD service have increased from US$91 million in 2018 to US$107 million in 2019 [1]. Taking Meituan platform as an example, about 4 million riders, 6.5 million restaurants, and 400 million customers are active on the platform in 2020 [2]. Overall, OFD service has high user demands and abundant market opportunities.

✉ Ling Wang
wangling@mail.tsinghua.edu.cn

Jie Zheng
j-zheng18@mails.tsinghua.edu.cn

Shengyao Wang
wsy06@aliyun.com

Jing-fang Chen
cjf17@mails.tsinghua.edu.cn

Xing Wang
wang-x17@mails.tsinghua.edu.cn

Haining Duan
duanhaining@meituan.com

Yile Liang
liangyile@meituan.com

Xuetao Ding
dingxuetao@meituan.com

[1] Department of Automation, Tsinghua University, Beijing 100084, China

[2] Meituan, Beijing 100102, China

Figure 1 shows the entire complex process of OFD platforms to deliver an order, which can be described as two phases: system scheduling and order delivering. In the system scheduling phase, the computing systems of platforms will decide which rider each order should be assigned to as well as plan the best route for each rider based on the collected global information, such as the information of all orders, riders, customers, and restaurants, as shown in Fig. 2. To be specific, after customers order their favorite food, platforms will inform corresponding restaurants to confirm as well as prepare food. Afterward, platforms will simulate the real delivery process, plan routes by heuristics, and evaluate the objectives of assigning each new order to each rider based on the planned route. Finally, platforms will dispatch orders to appropriate riders (usually with minimum total time) and inform them to confirm the delivery tasks.

After systems make satisfactory decisions, it enters the actual order delivering phase, where riders will head to restaurants to pick up food and ride to the locations of customers. Usually, riding is prohibited in or around buildings. Therefore, riders need to stop riding if they are near the customers' buildings, walk to customers' homes or workplaces, and hand food to customers. The total time of delivering an order can be described as the time from when the rider confirms the order to when the rider hands the food to the customer. It consists of three parts: (1) Pickup time: the time from when riders arrive at restaurants to when they leave. (2) Travel time: the time that riders cost on the route. (3) Service time: the time from when riders stop riding and enter customers' building to when they leave the customers' building and start riding. If the three parts can be estimated precisely in the system scheduling phase, platforms can accurately assess the total time of order assignment and make optimal decisions.

Among the three parts, the service time is of special significance, because it is required for many decision-makings in OFD service. The most important application is to assess the total time for the system scheduling. In addition, based on the statistics of service time, platforms can also adjust riders' wages, modulate delivery priorities of orders, and judge whether and where to install self-help dining cabinets, etc. In general, the service time of an order is unknown until it has been finished, but most decisions need to be made before the order delivery phase.

Therefore, predicting or modeling service time plays an important and indispensable part in OFD service. However, in the current scheduling process of Meituan, the service time is assumed as a deterministic value, which leads to inaccurate estimation of objectives and wrong decisions. Besides, to the best of our knowledge, many research works focus on predicting or modeling travel time [3] and food preparation time [4], but few on service time. Most researchers assume that the service time is already known [5, 6]. To fill up this gap, we intend to study the modeling and estimation of service time for OFD platforms. Usually, it is difficult to accurately estimate the service time of each order, because uncertainty is inevitable and pervasive throughout the service process. The service time will be prolonged if riders encounter various unexpected scenarios. For example, riders must stop riding and walk for a long distance in some villages that prohibit vehicles from entering. Besides, riders need to run up and down the stairs in some old residences without elevators. Even if there are elevators, riders sometimes must wait for a while to get on an elevator in office buildings with a high crowd flow. In addition, for serving the customers whose addresses are not clearly registered, it will take riders a lot of time to look for the particular position. These scenarios may happen all the time. However, it is hard to predict whether these scenarios will happen and how long it will take once any scenario happens for OFD platforms. Thus, predicting or modeling service time is a challenging task for OFD platforms.

There are many research works on the application of uncertain service time in scheduling problems. Some assumed that the service time distribution was known and simple, such as normal distribution, uniform distribution [7, 8], etc. However, service time distribution is much more complex and challenging to capture without priori knowledge in real life. Therefore, it usually cannot be assumed normally or uniformly distributed. Instead, some researchers employed fuzzy set theory to describe the uncertain service time [9], but this depiction is crude and cannot well highlight some tail features of actual situations. Therefore, it is necessary to build a precise model to reflect the fluctuation of service time. Besides, two additional requirements should also be satisfied. (1) The distribution should be in good form with analytical tractability. A suitable distribution form can effectively speed up the calculation process by countable additivity or mathematical analysis. (2) The distribution cannot occupy too much storage space. The storage resources of the OFD system are limited, because most of the resources are used to store necessary information on orders and riders. And it is also difficult for the system to transmit too much data in real time, especially during peak hours when the number of orders is enormous.

In this paper, we model the distribution of the uncertain service time as a Gaussian mixture model (GMM) and propose an intelligent estimation method to establish the model. This distribution estimation problem is then transformed into a clustering problem by regarding each cluster as a Gaussian component to determine the number of Gaussian components and other parameters of GMM. To solve the clustering problem, a hybrid estimation of distribution algorithm (HEDA) is presented to minimize distribution quality and simplicity. Solutions are generated by different sampling strategies. To be specific, in the initialization stage, the Chinese restaurant

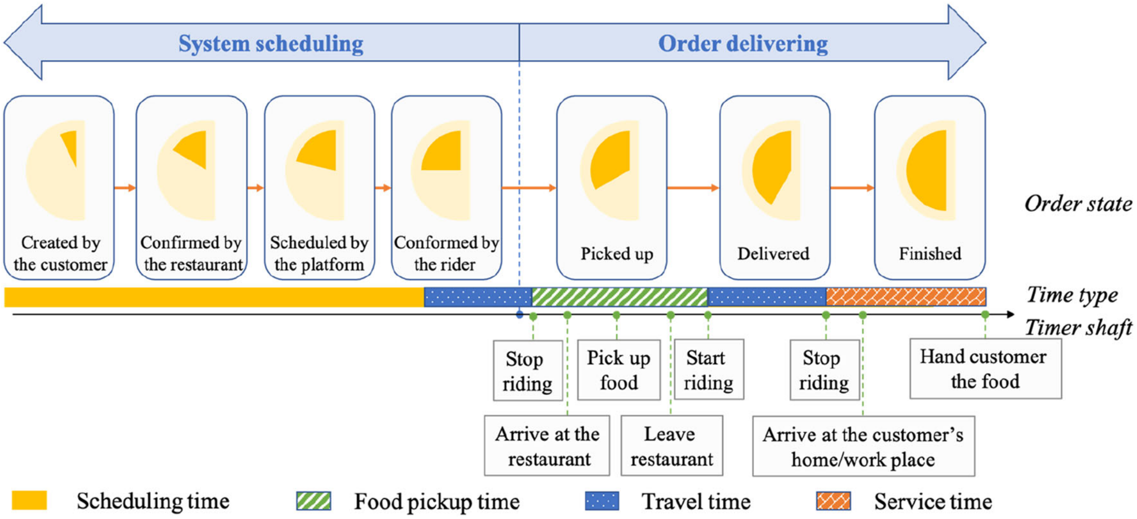

Fig. 1 The process to finish an order delivery task

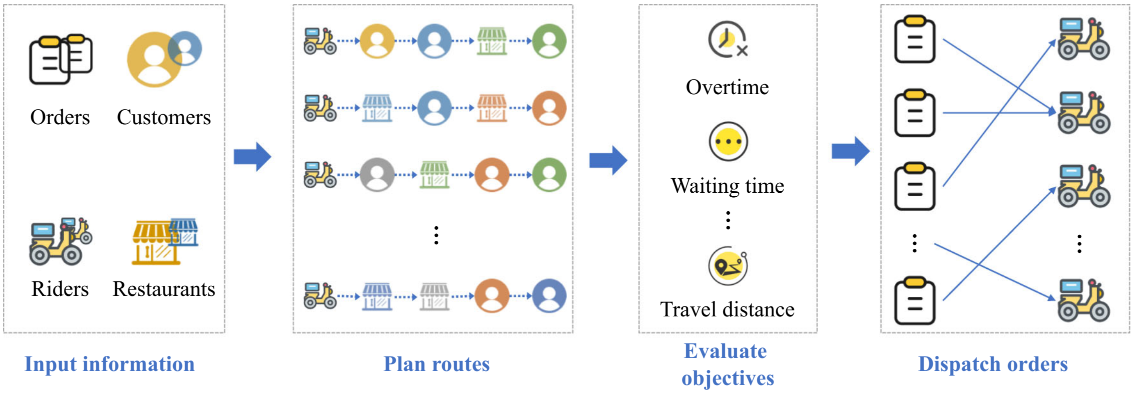

Fig. 2 Scheduling process of the OFD

process (CRP) is employed to sample the initial solutions with good quality; in the iterative stage, promising solutions are generated based on the sampling mechanism of the HEDA. By the design of the problem-specific probability matrix and cluster merging mechanism, the number of Gaussian components (or called clusters) can be learned dynamically during the searching process. Extensive offline experiments are conducted on real-world data and the results show the superiority of the proposed algorithm. In addition, to examine how the distribution model can help the decision-making for OFD platforms, we conduct an online A/B test on Meituan platform. The experimental results verify the practical value of the modeling and estimation of uncertain service time.

The remainder of the paper is organized as follows. Section "Literature review" gives the review of related literature. In Section "Problem description", the GMM estimation problem is described. Section "Algorithm" presents the HEDA in detail. Offline and online experimental results and analysis are provided in section "Experimental results". Finally, we end the paper with some conclusions in section "Conclusions".

Literature review

OFDP is an emerging field of research and related research works are lacking at present, let alone the uncertainty modeling and estimation in the OFDP. The research works on uncertain vehicle routing problem, which is one of the most related problems to OFDP, usually assumed the distribution of the service time was known with a simple form, such as a discrete probability distribution with finite support [10, 11], a uniform distribution [12, 13]. In addition, they did not quantify the actual benefits of introducing uncertainty into the problem for real-life application.

Bian et al. [14] presented a bus service time estimation model considering buses queueing for the curbside bus stop.

They employed the compound Poisson process to model the service time and calculated the probability of each scenario. However, it focuses on passengers' boarding and alighting process, as well as the buses queuing process at the stop area, whereas the scenario of OFD service is much more complex. As for the estimation of travel time, the major challenges are quite different from the service time, including the data sparsity of route segments, multiple combinations of route segments for a certain travel as well as the time limitation to answer users' queries [15]. The techniques of travel time estimation cannot be applied to service time estimation, although they are relatively mature. Therefore, it is necessary to propose an intelligent method to model the uncertain service time for OFD service.

According to [16], the class of mixture model densities can approximate any probability density function (PDF) if the number of mixture components of the model is taken to be sufficiently large. Thereinto, GMM, which is formed by multiple Gaussian components, is ubiquitously used for density estimation in many fields due to its analytical tractability and asymptotic properties [17]. The modeling of GMM can be divided into two types: first, the number of Gaussian components, denoted as K, is given or changeless; second, K is unknown or can be changed dynamically. For the former type, Redner and Walker [18] proposed a traditional Expectation-Maximum (EM) method to estimate the parameters of GMMs (including means, variances, and weight). It starts from an initial set of mixture parameters and generates a sequence of mixture parameters with increasing log-likelihood. However, this method was easy to get trapped in a local maximum and strongly dependent on the initial setting of parameters. To overcome the defect, Biernacki et al. [19] improved the EM algorithm using different initial strategies to find a good starting point. Besides, some swarm intelligent algorithms, such as genetic algorithm (GA) [20], particle swarm optimization [21], and differential evolution (DE) [22], were also employed for solving the problem due to their global exploration ability.

The latter type is more common in many practical applications where K is unknown and difficult to obtain at first. Biernacki and Govaert [23] assumed that K belonged to a set of values $\mathcal{K}$, and then estimated the parameters of the GMM corresponding to each possible K in $\mathcal{K}$. However, the method needs to be repeated card$(\mathcal{K})$ times to obtain the optimal solution which is too time-consuming, where card$(\cdot)$ means the number of elements in the set. Melnykov and Melnykov [24] presented a strategy based on the concentrations of neighbors to initialize mean vectors and determine K, and then used EM to estimate the final parameters. Nevertheless, due to the complexity of real-world data, the K determined by this method may not be optimal. Pernkopf and Bouchaffra [25] combined the GA with the EM method and used the minimum description length criterion to select the number of components. In addition, Bayesian learning algorithms have also been developed to estimate the GMM with an unknown K [26]. Bayesian-based approaches mainly include resampling methods [27] and variational inference [28]. The former is more accurate if given enough time but is too time-consuming in practical applications, whereas the latter is relatively quick but may lose some precision.

Table 1 The notations of the problem

Notation	Description
i	Index of data
k	Index of components
n	Number of available data
n_d	Number of unique data
K	Number of components
$K_{\max}$	Upper bound of component numbers
$\boldsymbol{t}$	Vector of service time, $\boldsymbol{t} = \{t_1, t_2, \ldots, t_n\}$
$\boldsymbol{t^u}$	Vector of unique service time, $\boldsymbol{t^u} = \left\{t_1^u, t_2^u, \ldots, t_{nd}^u\right\}$
f_k	Distribution of the kth Gaussian component
w_k	Weight of the kth Gaussian component
μ_k	Mean of the kth Gaussian component
σ_k^2	Variance of the kth Gaussian component
$\boldsymbol{\mu}$	Vector of all the means, $\boldsymbol{\mu} = \{\mu_1, \mu_2, \ldots, \mu_K\}$
$\boldsymbol{w}$	Vector of all the weights, $\boldsymbol{w} = \{w_1, w_2, \ldots, w_K\}$
$\boldsymbol{\sigma}^2$	Vector of all the variances, $\boldsymbol{\sigma}^2 = \left\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_K^2\right\}$
$\boldsymbol{\theta}$	Vector of all the parameters of the GMM,$\boldsymbol{\theta} = \left\{K, \boldsymbol{w}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2\right\}$
z	Vector of the clustering results of all the data,$z = \{z_1, z_2, \ldots, z_n\}$
f_d	The distribution of obtained GMM
f_s	The distribution of historical service time
W_d	Wasserstein distance between two distributions
λ	Penalty coefficient of K
ε	Objective function.$\varepsilon = W_\mathrm{d}(f_\mathrm{s}(\boldsymbol{t}), f_\mathrm{d}(\boldsymbol{t}; \boldsymbol{\theta})) + \lambda K$

Problem description

The notations of the problem are described in Table 1.

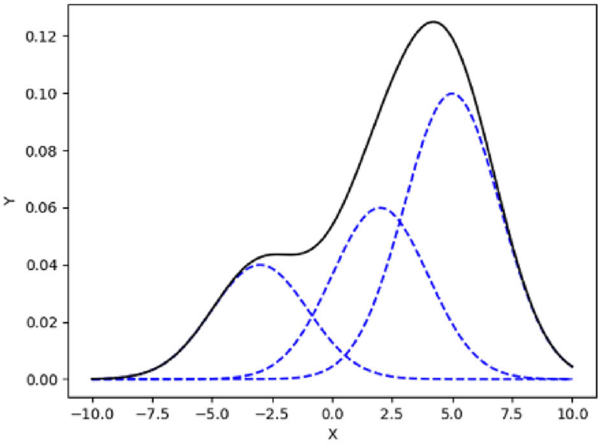

Fig. 3 An example of the GMM

A finite GMM $f(x;\boldsymbol{\theta})$ can be defined by a weighted sum of K components as follows:

$$f(x;\boldsymbol{\theta})=\sum_{k=1}^{K}w_k f_k\left(x;\mu_k,\sigma_k^2\right),\tag{1}$$

where w_k, μ_k, σ_k^2 are the weight, mean, and variance of the kth component, respectively. $\sum_{k=1}^{K}w_k=1$. $\boldsymbol{\theta}=\left\{K,\boldsymbol{w},\boldsymbol{\mu},\boldsymbol{\sigma}^2\right\}$ is the vector of all the parameters. $f_k\left(x;\mu_k,\sigma_k^2\right)$ is the following single Gaussian distribution function with parameter μ_k, σ_k^2:

$$f_k\left(x;\mu_k,\sigma_k^2\right)=\frac{1}{\sqrt{2\pi\sigma_k^2}}\exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right).\tag{2}$$

Figure 3 shows an example of the GMM with three components: the blue dotted lines are the distributions of different components and the full black line is the distribution of the mixture model.

We give our modeling of uncertain service time based on GMM as follows. Assume the input data are denoted as $\boldsymbol{t}=\{t_1,t_2,\ldots,t_n\}$, where n is the size of the dataset, and $t_i(i=1,\ldots,n)$ is the ith service time. Then, the real distribution of historical service time, denoted as f_{s}, can be approximated as the frequency function of the data

$$f_{\mathrm{s}}(t_i)=\sum_{j=1}^{n}I_{t_i=t_j}/n,\tag{3}$$

where $I_{a=b}$ is the following characteristic function:

$$I_{a=b}=\begin{cases}1, & \text{if } a=b\\ 0, & \text{else}\end{cases}.\tag{4}$$

Besides, the GMM obtained from historical service time, denoted as f_{d}, can be represented as follows:

$$f_{\mathrm{d}}(t_i;\boldsymbol{\theta})=\sum_{k=1}^{K}w_k f_k\left(t_i;\mu_k,\sigma_k^2\right).\tag{5}$$

Our goal is to determine the parameter vector $\boldsymbol{\theta}$ of f_{d} with the minimization of distribution quality and simplicity synchronously, where the distribution quality means the similarity between the estimated GMM and the real distribution of historical data, and the simplicity means the number of the GMM parameters. Kullback–Leibler divergence is commonly used in the previous works [29] to measure the similarity of two distributions. However, it has obvious weaknesses, such as asymmetry and unboundedness [30]. Instead, Wasserstein distance, presented by [31], not only has none of these disadvantages but also can measure the minimum average distance required to move the data from one distribution to another. Therefore, Wasserstein distance is employed in this paper to construct the objective function ε, which is calculated as follows:

$$\min_{\boldsymbol{\theta}}\varepsilon=W_{\mathrm{d}}(f_{\mathrm{s}}(\boldsymbol{t}),f_{\mathrm{d}}(\boldsymbol{t};\boldsymbol{\theta}))+\lambda K\tag{6}$$

$$W_{\mathrm{d}}(f_{\mathrm{s}}(\boldsymbol{t}),f_{\mathrm{d}}(\boldsymbol{t};\boldsymbol{\theta}))=\sum_{x=\min\boldsymbol{t}}^{\max\boldsymbol{t}}|f_{\mathrm{s}}(x)-f_{\mathrm{d}}(x;\boldsymbol{\theta})|,\tag{7}$$

where λ is a given weight coefficient, $W_{\mathrm{d}}(f_{\mathrm{s}},f_{\mathrm{d}})$ is the Wasserstein distance between f_{s} and f_{d} under L1-norm, and λK is a penalty to guarantee the simplicity.

This minimization problem has infinite solution space and is hard to obtain the analytical solution in a short time. Although several methods have been proposed to estimate the GMM with an unknown K, they are either poor in quality [23] or too time-consuming for Meituan platform due to the huge amount of data [27]. To well solve the problem, we transform it into a clustering problem. Each data can be assigned to a certain cluster, where a cluster represents a component of the GMM. Based on the clustering results $\boldsymbol{z}=\{z_1,z_2,\ldots,z_n\}$ where $z_i=k$ means that the ith data are assigned to cluster k, K can be determined by the number of clusters, and the parameters $\boldsymbol{w}$, $\boldsymbol{\mu}$, $\boldsymbol{\sigma}^2$ can be easily derived by calculating the maximum-likelihood estimation for each cluster as follows:

$$K=\max(z_1,z_2,\ldots,z_n)\tag{8}$$

$$\mu_k=\sum_{j=1}^{n}I_{z_i=k}t_i\Big/\sum_{i}^{n}I_{z_i=k},\ k=1,\ldots,K\tag{9}$$

$$\sigma_k^2=\sum_{i=1}^{n}I_{z_i=k}(t_i-\mu_k)^2\Big/\sum_{i}^{n}I_{z_i=k},\ k=1,2,\ldots,K.\tag{10}$$

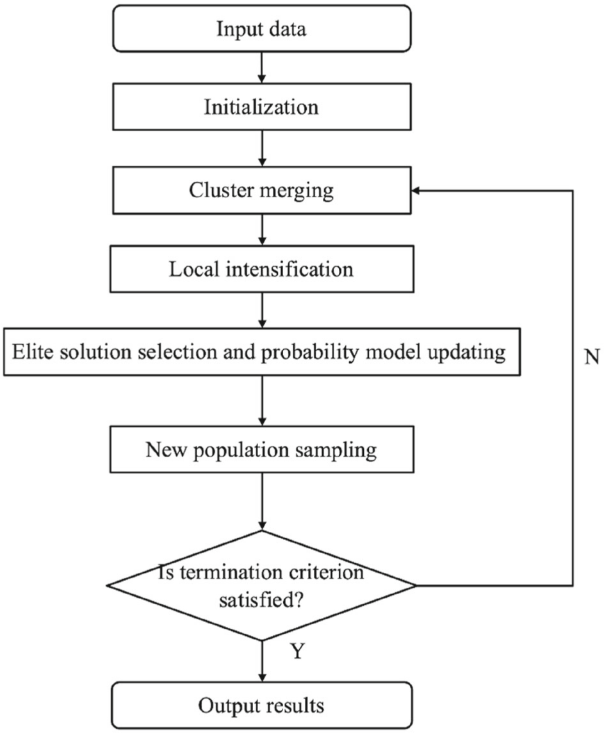

Fig. 4 The framework of the HEDA

In a word, the decision variables are transformed as z after defining the clustering problem. Then, the objective function will be transformed into

$$\min_{z} \varepsilon = W_{\mathrm{d}}(f_{\mathrm{s}}(\boldsymbol{t}),\ f_{\mathrm{d}}(\boldsymbol{t};\ z)) + \lambda K, \tag{11}$$

where $f_{\mathrm{d}}(\boldsymbol{t};\ z)$ can be calculated by Eq. (5) and Eqs. (8)–(10).

Algorithm

Estimation of the distribution algorithm (EDA) is a swarm intelligent algorithm of good global exploration capacity [32, 33]. It is a statistical learning method that explores potential search space by building and sampling an explicit probability model of promising elite individuals. It has been employed in many fields, such as factory production scheduling [34], financial decision-making [35], green energy technology [36], and so on. In this paper, we design a HEDA to build the GMM by the cooperation of the EDA and the EM method. Besides, to obtain the initial solutions with a certain quality, the Chinese restaurant process (CRP) [37], one typical model of the Dirichlet process [38], is employed for initial clustering. The overall framework is shown in Fig. 4. We will present the algorithm with data pre-processing, encoding and decoding methods, initialization, cluster merging, local intensification, elite solutions selection, probability model updating, and new population sampling.

Encoding and decoding

The HEDA is proposed as a clustering method where the encoding represents the clustering results of each data, and the decoding is to calculate the distribution parameters according to maximum-likelihood estimation. To be specific, solutions can be represented by the clustering vector z of the length n. However, since the amount of input data is huge, this method will occupy too much space. Instead, only the clustering results of unique data are stored to save space, because there is a lot of duplication in data. To be specific, a matrix PM of size $K_{\max} \times n_{\mathrm{d}}$ is used to represent solutions, where $K_{\max}$ is the given maximum number of components due to the storage limitation, and n_{d} is the number of unduplicated data. Each element PM_{ki} of PM is defined as the count that the ith number is classified into cluster k as follows. Besides, $\boldsymbol{C} = \left\{c_1, c_2, \ldots, c_{n_d}\right\}$ is the count vector of the number of each unique value in the data, where $\sum_{k=1}^{K_{\max}} p_{ik} = c_i$ must be satisfied

$$\mathrm{PM} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n_d} \\ p_{21} & p_{22} & \cdots & p_{2n_d} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K_{\max}1} & p_{K_{\max}2} & \cdots & p_{K_{\max}n_d} \end{bmatrix}. \tag{12}$$

Decoding is to obtain $\boldsymbol{\theta}$ via encoding, which is adjusted on the basis of Eqs. (8)–(10). Before calculating $\boldsymbol{\theta}$, there may be some same or related rows in the PM, which can be merged to reduce the number of components (clusters). The most basic idea is to do elementary row transformation to find a normal form of PM, and set the rank of the transformed PM as K. This method can minimize K of a certain PM, but is too time-consuming for large matrixes. Therefore, we only operate some simple row transformations to merge identical rows and turn the PM into $\left[\mathrm{PM}', 0\right]^T$. Then, $\boldsymbol{\theta}$ can be calculated as follows:

$$K = N_r\left(\mathrm{PM}'\right) \tag{13}$$

$$\mu_k = \sum_{i=1}^{n_d}\left(t_i^u \times \mathrm{PM}'_{ik}\right) \Big/ \sum_{i=1}^{n_d} \mathrm{PM}'_{ik},\quad k = 1, 2, \ldots, K \tag{14}$$

$$\sigma_k^2 = \sum_{i=1}^{n_d}\left(t_i^u - \mu_k\right)^2 \times \mathrm{PM}'_{ik} \Big/ \sum_{i=1}^{n_d} \mathrm{PM}'_{ik},\quad k=1, 2, \ldots, K \tag{15}$$

$$w_k = \sum_{i=1}^{n_d} \mathrm{PM}'_{ik} \Big/ n,\quad k = 1, 2, \ldots, K, \tag{16}$$

where $N_r\left(\mathrm{PM}'\right)$ is the number of rows of PM′.

For example, given the input service time as $\boldsymbol{t} = \{2, 3, 5, 3, 2, 6, 2, 6\}$ and $K_{\max}$ as 3, then the unique data set $\boldsymbol{t^u}$ is $\{2, 3, 5, 6\}$ with $n_\mathrm{d} = 4$ and $\boldsymbol{C} = \{3, 2, 1, 2\}$, and a possible realization of PM can be

$$\mathrm{PM} = \begin{bmatrix} 1\ 1\ 0\ 1 \\ 1\ 1\ 0\ 1 \\ 1\ 0\ 1\ 0 \end{bmatrix}. \tag{17}$$

It can be transformed to $\left[\mathrm{PM}', 0\right]^T$ as follows, where $\mathrm{PM}' = \begin{bmatrix} 2\ 2\ 0\ 2 \\ 1\ 0\ 1\ 0 \end{bmatrix}$.

$$\mathrm{PM} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \xrightarrow[\substack{\text{Add Row 2 to Row 1} \\ \text{Set Row 2 to zero}}]{\textit{Merge}} \begin{bmatrix} 2 & 2 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \xrightarrow[\text{Move Row 2 to the end}]{\textit{Move}} \begin{bmatrix} 2 & 2 & 0 & 2 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathrm{PM}' \\ \mathbf{0} \end{bmatrix} \tag{18}$$

Then, the parameters can be calculated as $K = 2$, $\mu_1 = 2 \times (2+3+6)/6 \approx 3.6667$, $\mu_2 = (2+5)/2 = 3.5$, $\sigma_1^2 = 2 \times \left[(2-3.6667)^2 + (3-3.6667)^2 + (6-3.6667)^2\right]/6 \approx 2.8889$, $\sigma_2^2 = \left[(2-3.5)^2 + (5-3.5)^2\right]/2 = 2.25$, $w_1 = 6/8 = 0.75$, $w_2 = 2/8 = 0.25$. Therefore, $\boldsymbol{\mu} = \{3.6667, 3.5\}$, $\boldsymbol{\sigma}^2 = \{2.8889, 2.25\}$, $\boldsymbol{w} = \{0.75, 0.25\}$, $\boldsymbol{\theta} = \{2, 0.75, 0.25, 3.6667, 3.5, 2.8889, 2.25\}$.

In general, our encoding can use a small storage space to represent the entire solution space, and our decoding method can quickly obtain $\boldsymbol{\theta}$ as well as reduce K.

Initialization

The CRP is a typical discrete-time stochastic process in probability theory, and has been widely employed to represent uncertainty over the number of components in a mixture model [37]. It builds a distribution on partitions of integers to describe the process where n customers sit down in a Chinese restaurant with an infinite number of tables. Definitely, the first customer sits at the first table. For the ith subsequent customer, the tendency to choose the table obeys the following distribution:

$$p(T_i = k|T_1, T_2, \ldots, T_{i-1}) = \frac{n_k}{\gamma + i - 1} \tag{19}$$

$$p(T_i = K+1|T_1, T_2, \ldots, T_{i-1}) = \frac{\gamma}{\gamma + i - 1}, \tag{20}$$

where n_k is the number of previous customers sitting at table k, T_i is the chosen table of customer i, K is the number of occupied tables, and γ is a given parameter that describes the dispersion degree of the distribution.

CRP has good clustering properties, and can be employed to sample initial solutions for our problem. To ensure the diversity of the population, a shuffled index vector will be used to access service time for each solution, denoted as $\boldsymbol{S} = \{s_1, s_2, \ldots, s_n\}$, where $1 \le s_1, s_2, \ldots, s_n \le n_d$, and the constraint that the number of elements in $\boldsymbol{S}$ that are equal to i is c_i, for $i = 1, 2, \ldots, n_d$ should also be satisfied. Similarly, t_{s1} belongs to the first cluster, denoted as $z_{s_i} = 1$, and $\mathrm{pm}_{s_1,1} = 1$. For the ith subsequent data, since the number of components should not be larger than $K_{\max}$, the distribution is transformed as follows:

$$p\left(z_{s_i} = k|z_{s_1}, z_{s_2}, \ldots, z_{s_{i-1}}, K < K_{\max}\right) = \frac{\sum_{i=1}^{n_d} \mathrm{PM}'_{ik}}{\gamma + i - 1} \tag{21}$$

$$p\left(z_{s_i}=K+1|z_{s_1}, z_{s_2}, \ldots, z_{s_{i-1}}, K < K_{\max}\right)=\frac{\gamma}{\gamma + i - 1} \tag{22}$$

$$p\left(z_{s_i} = k|z_{s_1}, z_{s_2}, \ldots, z_{s_{i-1}}, K = K_{\max}\right) = \frac{\sum_{i=1}^{n_d} \mathrm{PM}'_{ik}}{\sum_{k=1}^{K} \sum_{i=1}^{n_d} \mathrm{PM}'_{ik}}. \tag{23}$$

By sampling the distribution, we can divide the ith data into a specific cluster k^*, and then, $\mathrm{pm}'_{s_i,k^*} = \mathrm{pm}'_{s_i,k^*} + 1$. If it is assigned to a new cluster, then $K = K + 1$. Repeat the above sampling process and we will obtain the initial population which is consisted of Pn individuals.

In addition to the initial population, the probability matrix should also be initialized. It is the core of the EDA, which affects the whole evolution and updating process of the population. Denote Q as the probability matrix of size $K_{\max} \times n_d$. Each element $q_{ki} \in Q$ represents the probability that t_i^u belongs to cluster k. In this part, Q is initialized uniformly to guarantee sufficient randomness for population diversity. That is, $q_{ki} = 1/K_{\max}$.

Cluster merging

To further reduce the number of Gaussian components, a cluster merging strategy is designed. To be specific, some clusters with very few elements may be generated during the sampling process of the CRP or HEDA. These clusters are with small weights and have little effect on the overall distribution. Therefore, to reduce the length of $\boldsymbol{\theta}$, we discard these clusters and redistribute the elements to other clusters with large weights. The pseudo-code of the cluster merging is given as Algorithm 1. The threshold of the smallest number *thres* of elements is $n/2K_{\max}$.

Algorithm 1: Cluster merging

```
$K^{rmv} \leftarrow \emptyset$: the index set of the removed cluster
For $k = 1$ to $K$
    If $\sum_{i=1}^{n_d} pm_{ki} < thres$
        $K^{rmv} = K^{rmv} \cup k$
    End if
End for
For $k$ in $K^{rmv}$
    For $i = 1$ to $n_d$
        For $j = 1$ to $pm_{ki} \times c_i$
            Assign data $t_j$ to a randomly selected $k_{new}$, $1 \le k_{new} \le K$ and $k_{new} \notin K^{rmv}$
            $pm_{ki} = pm_{ki} - 1$
            $pm_{k_{new}i} = pm_{k_{new}i} + 1$
        End for
    End for
End for
Repair $PM$ by row transformations
Return $PM$.
```

Local intensification

The EDA has a strong global search capacity, but a relatively weak local searchability, while the EM algorithm is on the contrary. Therefore, we employ the EM algorithm as the local intensification operator to balance the exploration and exploitation of the HEDA.

The parameter vector of the best solution obtained during the current iteration is set as the initial solution of the EM algorithm. Then, by alternating E-step and M-step, the parameter vector will be improved until a certain search depth *ls* is reached. The E-step and M-step are defined as follows.

E-step: Given the parameter vector from the previous iteration of the EM algorithm, the posterior probability can be computed as

$$\mathrm{PM}_{k,i}^{g} = c_i w_{k,g} f_k\left(t_i^u|\mu_{k,g}, \sigma_{k,g}^2\right) / \sum_{l=1}^{K} w_{k,g} f_l \times \left(t_i^u|\mu_{l,g-1}, \sigma_{l,g-1}^2\right), \tag{24}$$

where $w_{k,g}$, $\mu_{k,g}$ and $\sigma_{k,g}^2$ denotes the weight, mean, and variance parameters of kth Gaussian component in gth iteration.

M-step: Given the posterior probabilities $\mathrm{PM}_{k,i}$ obtained in the E-step, the parameters $w_{k,g}$, $\mu_{k,g}$, and $\sigma_{k,g}^2$ can be calculated according to Eqs. (14)–(16).

Updating probability model

The probability model is updated by elite solutions, which is defined as the first e_n solutions with minimal objective value. To better learn about the probability information, a sequence-dependent weight strategy is designed. To be specific, we will sort the elite solutions by their objective, denoted as $\text{Elite} = \left\{E_{(1)}, \ldots, E_{(e_n)}\right\}$, and the better ones will be assigned with a higher weight. The weight $w'_{(e)}$ of $E_{(e)}$ is set as

$$w'_{(e)} = \frac{e_n + 1 - e}{\sum_{j=1}^{e_n} j}. \tag{25}$$

Then Q is updated as follows:

$$q_{ki} = (1-\alpha)q_{ki} + \alpha p_{ki}(\text{Elite}) \tag{26}$$

$$p_{ki}(\text{Elite}) = \sum_{e=1}^{e_n} w'_{(e)} p_{E_{(e)}}(i, k)/c_i, \tag{27}$$

where $\alpha \in (0, 1)$ is the learning rate. Besides, Q should be normalized by column to ensure feasibility.

Sampling mechanism

For each iteration, new individuals are sampled from the probability model Q. To ensure the population diversity, a

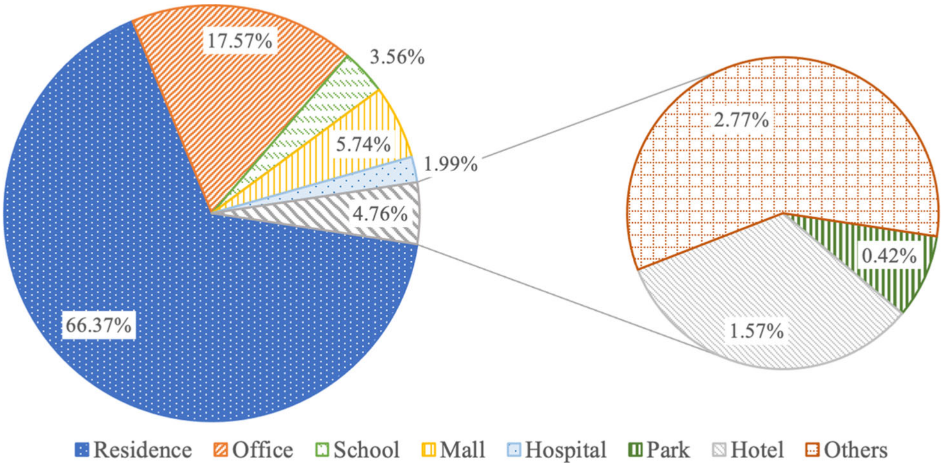

Fig. 5 The distribution of the instances on AOI types

temporary probability matrix Q' is generated by adding disturbances to Q, and is normalized by column to guarantee the feasibility. To ensure the quality of the solution, we narrow the range of disturbance, which is set as the maximum probability of a certain column as follows:

$$q'_{ki} = q_{ki} + \delta p, \forall i, k, \tag{28}$$

where δp is a random number between 0 and $\max(q_{i1}, \ldots, q_{iK})$

$$\text{PM}_{ki} = q'_{ki} \times c_i / \sum_{k=1}^{K} q'_{ki}, \forall i, k. \tag{29}$$

Experimental results

In this section, extensive offline and online tests are conducted. The offline experiments are carried out by comparing the HEDA with several existing algorithms on real-world datasets from the Meituan platform to validate the estimation effectiveness of GMM. The online A/B tests are implemented by employing the GMM obtained by the HEDA on the real-world scheduling system to validate the efficacy of the utilization of stochastic models. The details of the experiments are elaborated as follows.

Offline experiment

Experimental settings

Instance generation In this part, we will introduce the instance generation of the offline experiment. The instances of the offline experiment are the data set of service time $\boldsymbol{t} = \{t_1, t_2, \ldots, t_n\}$ in different areas and time periods. Service time differs greatly in space and time. Specifically, because residential areas usually have a more complex building environment and may not allow riders to pass, the service time in residential areas maybe be longer than that of office buildings. Besides, due to factors such as the passenger flow of elevators or other passing facilities, the service time during the peak hours is regularly longer than idle hours. Therefore, we distinguish the distribution of uncertain service time under different areas of interest (AOI) (each residence, office, school, mall, hospital, and so on) and periods (peak hours and idle hours). Each city can be divided into many AOIs. For example, in Beijing, there are more than 3000 schools alone. That is, tens of thousands of GMMs need to be constructed for cities throughout the whole country, which forces us to construct accurate GMMs with parameters vectors as short as possible. By collocating 1 month's historical data from multiple cities, a total of 29,028 instances are obtained. The distributions of the instances divided by AOI types and periods are shown in Figs. 5 and 6, respectively.

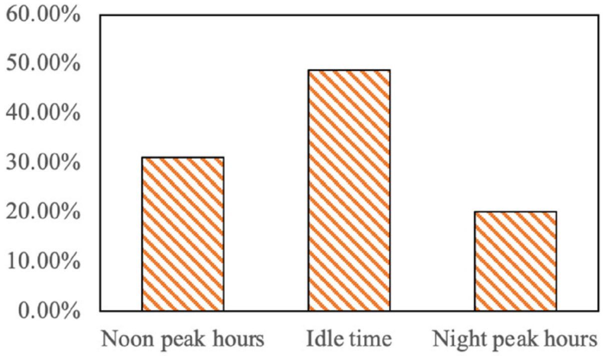

Fig. 6 The distribution of the instances on periods

Due to the possibility of extremely abnormal situations, some data may deviate far from the majority in the collected dataset of different AOI and periods. The analysis and statistics of the abnormal data are of little significance, so we need to clean the data first to ensure the validity of the input. After

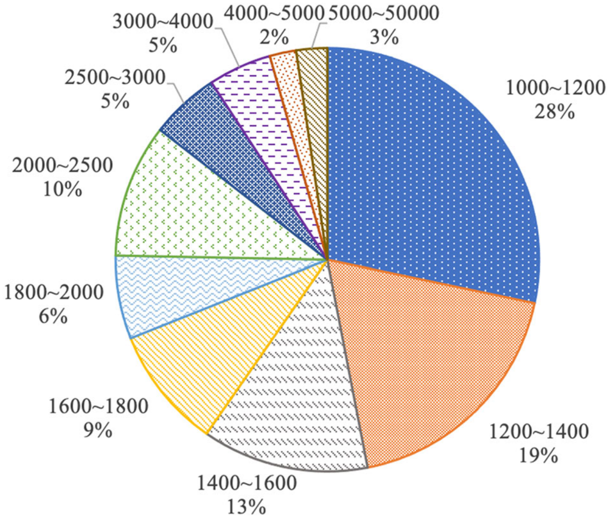

Fig. 7 The distribution of the instances on the amount of data

Table 2 Combinations of parameter values

Parameters	1	2	3	4
Pn	50	100	150	200
e_n	5	10	15	20
α	0.2	0.4	0.6	0.8

removing the null values and singularities, we will retain the 95% confidence data as the input. Besides, in light of the amount of data in each instance which infects the computational complexity of building the GMMs, we also divide these instances into ten groups as shown in Fig. 7, where the ranges of the data amount of each group are marked around the pie.

Table 3 Orthogonal array and RV values

Pn	e_n	α	RV
1	1	1	0.113443
1	2	2	0.110163
1	3	3	0.109773
1	4	4	0.111208
2	1	2	0.110450
2	2	1	0.112905
2	3	4	0.110893
2	4	3	0.111915
3	1	3	0.112769
3	2	4	0.111772
3	3	1	0.113471
3	4	2	0.112926
4	1	4	0.114224
4	2	3	0.109790
4	3	2	0.111676
4	4	1	0.116223

Table 4 Response value and rank of each parameter

Level	Pn	e_n	α
1	0.111147	0.112722	0.114010
2	0.111541	0.111157	0.111304
3	0.112735	0.111453	0.111062
4	0.112978	0.113068	0.112024
Delta	0.001831	0.001911	0.002949
Rank	3	2	1

Parameters The HEDA contains three key parameters: the population size Pn, the learning rate α of Q, and the number of elite solutions e_n. To investigate the effect of these parameters on the performance of the algorithm, we implement the Taguchi method of design of the experiment using a moderate scale instance in group 7. Four levels of each parameter are set as shown in Table 2. According to the number of parameters and levels, an orthogonal table L16(4^3) is generated with 16 combinations of parameter values in total. For each combination, we run the HEDA 20 times independently to obtain the average ε, which is set as the response value (RV). The smaller the RV, the better the performance of the parameter combination. The orthogonal array and RVs are listed in Table 3. The significant rank of each parameter is listed in Table 4. Besides, the trend of each factor is shown in Fig. 8. It can be seen that α is the most significant parameter. A large value of α may lead to premature convergence, while a small value of α may lead to slow convergence. Besides, e_n ranks the second. A proper e_n can help the algorithm update the probability model reasonably. As for Pn, it ranks third. If Pn is too large, it will be too time-consuming for one iteration and lead to slow convergence. However, if Pn is too small, the solution space cannot be sampled sufficiently. According to the above analysis, the parameters of the HEDA are suggested as Pn $= 50$, $e_n = 10$, and $\alpha = 0.6$, which will be used in the following experiments. Besides, λ is empirically set as 0.1.

Comparison algorithms We compare our HEDA with the following three other methods.

- Nested-EM [39], which was proposed recently to build GMMs with an unknown K.

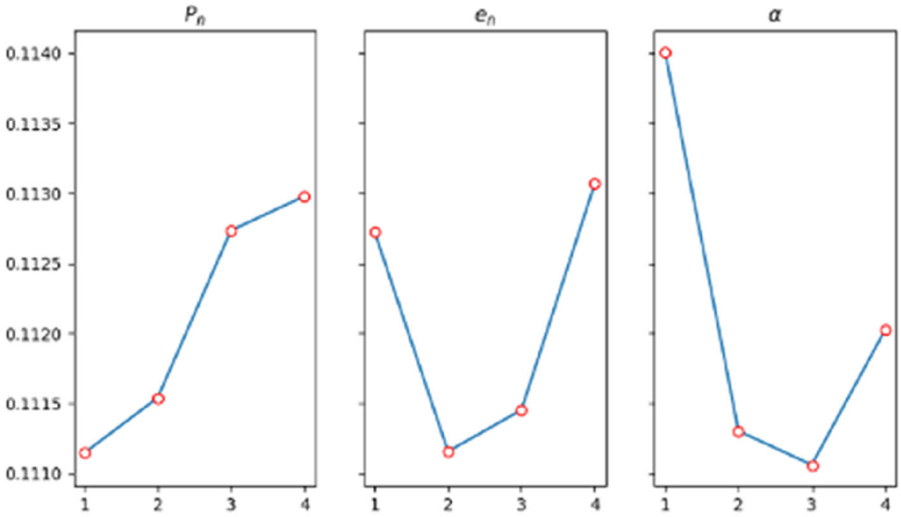

Fig. 8 Factor level trend

Table 5 Results of RPD and p value

Group	HEDA	Nested-EM		DE		HEDA-nls	
	RPD	RPD	p value	RPD	p value	RPD	p value
$1000 \le n \le 1200$	2.03	5.20	0.02	35.11	0.00	11.85	0.00
$1200 < n \le 1400$	1.73	5.96	0.02	41.45	0.00	13.55	0.00
$1400 < n \le 1600$	1.81	5.97	0.02	41.82	0.00	13.95	0.00
$1600 < n \le 1800$	1.87	5.91	0.02	41.53	0.00	13.79	0.00
$1800 < n \le 2000$	1.61	5.96	0.03	41.29	0.00	13.60	0.00
$2000 < n \le 2500$	1.76	5.86	0.02	41.49	0.00	13.68	0.00
$2500 < n \le 3000$	1.72	5.85	0.03	41.36	0.00	13.46	0.00
$3000 < n \le 4000$	1.63	5.72	0.03	40.94	0.00	13.46	0.00
$4000 < n \le 5000$	1.83	5.52	0.02	40.92	0.00	14.16	0.00
$5000 < n \le 50,000$	1.67	5.78	0.02	40.98	0.00	13.67	0.00

- Differential evolution (DE) algorithm [22], which is proposed to solve the GMM with a priori number of components. For a fair comparison, we set the priori number of components as $K_{\max}$ and adapt the cluster merging operator to the algorithm to reduce K.
- The K-means algorithm [40], which is a typical and most known clustering method. We also compare our proposed HEDA to this well-known clustering method, since we transmit our distribution estimating problem into a clustering problem.

Besides, to demonstrate the effectiveness of the local intensification, we compare the HEDA with HEDA_nls (HEDA without local intensification). The termination criterion for each algorithm is the same evaluation times. All the algorithms are coded in Java and run on the servers of Meituan.

Evaluation metric In this paper, relative percentage deviation (RPD) and proportion of best times (PBT) are employed to evaluate the performance

$$\mathrm{RPD}\left(\mathrm{alg}_i\right) = \frac{\mathrm{alg}_i - \mathrm{bst}_i}{\mathrm{bst}_i} \times 100 \tag{30}$$

$$\mathrm{PBT}\left(\mathrm{alg}_i\right) = \sum_{i=1}^{N} I_{\mathrm{alg}_i = \mathrm{bst}_i} / N \times 100, \tag{31}$$

where alg_i is the objective value of a certain algorithm and bst_i is the best objective value obtained by all the compared algorithms for the ith instance. A smaller RPD indicates a better performance. PBT represents the proportion of the times that an algorithm obtains the best solutions in the group with N instances. A larger PBT indicates a better performance.

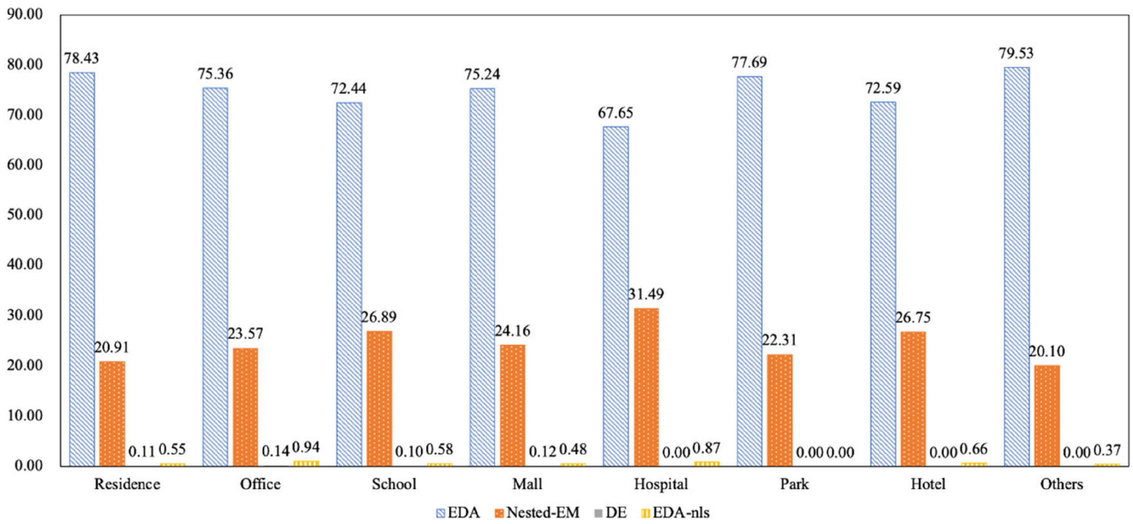

Fig. 9 The PBT on different AOI types

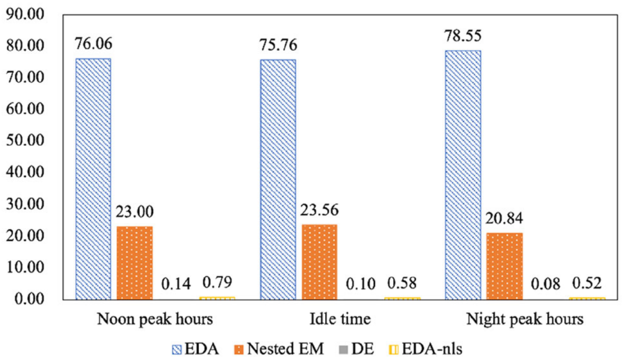

Fig. 10 The PBT on different periods

Experimental results

The results of the RPD on different groups are listed in Table 5. It can be seen that the HEDA is better than other algorithms in all groups of instances, which indicates the superiority of the HEDA. And the difference between the RPDs of HEDA and other algorithms, especially the nested EM in Group 1 is significantly smaller than in other groups. It may be because the instances of Group 1 are with the smallest problem scale and are easier to be solved. Besides, by comparing the HEDA and HEDA_nls, it can be concluded that the local intensification can effectively improve the performance of the algorithm. In addition, the nonparametric tests (Kruskal–Wallis tests) with 95% confidence interval are carried out for each pair of RPDs.

Besides, the results of PBT on different AOI types are shown in Fig. 9. It can be seen that the proposed HEDA is the best on most AOI types, especially on the residence, which occupies 66.37% of all the instances.

From Fig. 10, it can be seen that the PBT of HEDA is close to 80% at any period. Therefore, we can conclude that the HEDA is superior on different AOI types and periods to other compared algorithms.

Online A/B test

Experiment settings

Settings of policy and the A/B test To quantify the benefits of introducing the uncertainty of service time into the OFDP for real-life application, we compare the deterministic policy and stochastic policy to dispatch orders on Meituan platform. The two policies are shown in Fig. 11. And the specific details of route planning and order dispatching algorithms can be found in reference [41]

- Deterministic policy, Fig. 11a. It is the current scheduling policy of Meituan, where the service time is assumed as a deterministic value and is obtained according to the average value of historical service time, leading to deterministic objective values for order dispatching.
- Stochastic policy, Fig. 11b. We employ the GMMs of stochastic service time generated by the HEDA. For utilization, multiple scenarios are sampled from the GMMs via the Monte Carlo sampling method to calculate the objective values of solutions. And the expected objective will be employed to guide the order dispatching.

Then, online A/B test is employed to compare these two policies. The A/B test is conducted in Zhuhai, Guangdong Province, China. Similar to the test settings by Chen et al. [42], we divide the areas in this city into two similar regions:

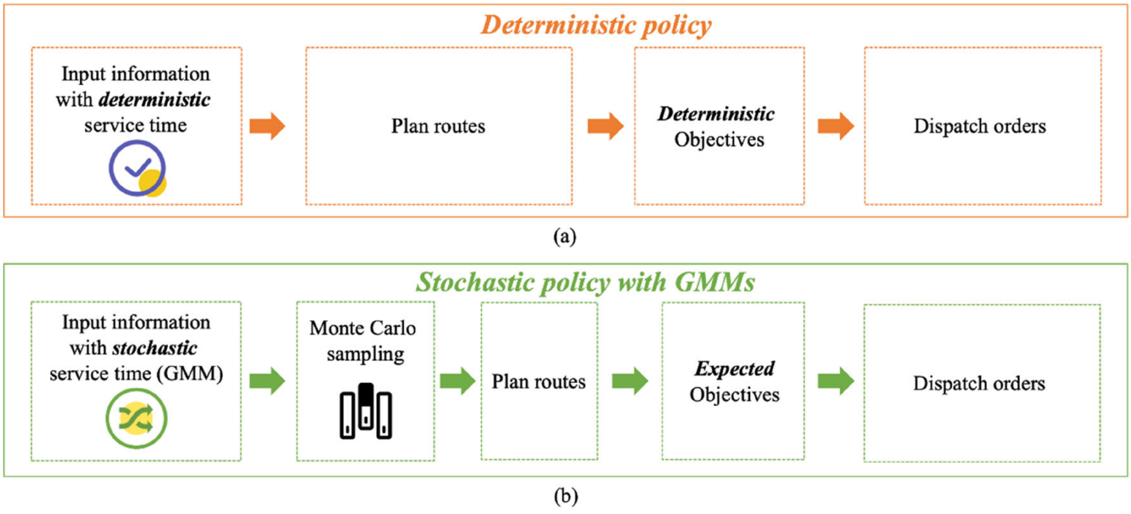

Fig. 11 The framework of scheduling process in Meituan

an experimental region (E-region) and a control region (C-region). And the input of the online experiment includes the whole information of riders, orders, restaurants, as well as the distributions or deterministic values of service time. For a fair comparison, we guarantee that there are no significant differences between the two regions before the test. The observation metrics of the E-region and C-region are as close as possible during a period (denoted as comparison dates) when dividing areas. Afterward, we employ the deterministic method in the C-region and the stochastic method in the E-region to dispatch orders relatively. These methods are run for a period of time (denoted as experimental dates), which lasts for the same duration as the comparison dates.

Observation metric For the online test, several observation metrics are adopted to evaluate customer satisfaction and delivery efficiency as follows.

(a) Customer satisfaction-based metric.

- 5-Min punctual rate (PR_5):

$$\mathrm{PR}_5 = \frac{\left|\hat{O}_{5-}\right|}{|O|} \times 100\%. \tag{32}$$

- 15-Min overtime rate (OR_{15}):

$$\mathrm{OR}_{15} = \frac{\left|\hat{O}_{15^+}\right|}{|O|} \times 100\%. \tag{33}$$

- Proportion of orders delivered over 55 min (PP_{55}):

$$\mathrm{PP}_{55} = \frac{\left|\hat{O}_{55^+}\right|}{|O|} \times 100\%. \tag{34}$$

(b) Delivery efficiency-based metric.

- Average online delivery time (AODT) (unit: minute):

$$\mathrm{AODT} = \frac{1}{|O|} \sum_{o(i) \in O} \left(\widehat{\mathrm{DT}}_i - \mathrm{RT}_i\right). \tag{35}$$

- Area time efficiency (ATE):

$$\mathrm{ATE} = \frac{|O|}{\hat{t}_{\mathrm{del}}}, \tag{36}$$

where O is the set of all delivered orders during a day in a certain area, $\hat{O}_{5^-}$ is the set of orders that are delivered without delay or delayed within 5 min, O_{15^+} is the set of orders that are delayed for more than 15 min, $\widehat{\mathrm{DT}}_i$ is the time when o_i is delivered to the customer, RT_i is the time when o_i is accepted by the platform, and $\hat{t}_{\mathrm{del}}$ is the total time spent by all the riders to deliver orders. The units of AODT and $\hat{t}_{\mathrm{del}}$ are both minutes. The notation with a "^" means that it is observed online. The PR_5, PP_{55}, ATE are the bigger the better, and the OR_{15}, AODT are the smaller the better.

Experiment results

Although the ideal situation is that the observation metrics of the E-region and C-region are exactly the same before the test, it is difficult to achieve this situation. Therefore, to mitigate the effect caused by the deviation, the following true difference Δ_{tru} is introduced as the comparison metric.

We believe that the difference of each metric between the E-region and C-region during the comparison dates, denoted as inherent deviation (Δ_{inh}), can represent the essential difference between the two regions. Besides, the difference during the experiment dates, denoted as the experimental difference

Table 6 The results of the online A/B test

Metric	Comparison dates			Experimental dates			Δ_{tru}
	C-region	E-region	Δ_{inh}	C-region	E-region	Δ_{exp}	
PR_5	98.68%	98.66%	− 0.02%	97.14%	97.47%	0.33%	0.35%
OR_{15}	0.21%	0.25%	0.04%	0.72%	0.55%	− 0.17%	− 0.21%
PP_{55}	0.67%	0.65%	− 0.02%	4.02%	3.24%	− 0.77%	− 0.75%
AODT	25.7542	26.1753	1.64%	30.5559	30.2188	− 1.10%	− 2.74%
ATE	4.8523	4.9164	1.32%	5.2246	5.3660	2.71%	1.39%

(Δ_{exp}), should include this inherent deviation and the influence caused by different methods. Then, $\Delta_{\text{tru}} = \Delta_{\text{exp}} - \Delta_{\text{inh}}$ can be regarded as the true difference of different methods by excluding the influence of inherent deviation.

The results of the online A/B test are shown in Table 6, where the bold value of Δ_{tru} means that the corresponding metric is improved by employing the stochastic service time model in the dispatching system. It can be seen that our new GMM-based policy can improve all the metrics compared to the current policy. Roughly speaking, if we extend this policy to the whole country, we can reduce about 30,000 severely-overtime orders a day which can greatly improve the customers' satisfaction. Besides, about 50 s can be saved on average for delivering an order, which indicates an improvement in rider efficiency. Therefore, we can conclude that accurate description and application of uncertain service time can effectively improve customer satisfaction and delivery efficiency for OFD applications.

Conclusions

This paper focused on the modeling of uncertain stochastic time in OFD applications. An HEDA was presented to build GMMs with an unknown component number of each AOI under different scenarios. Extensiveoffline numerical tests of real-world data showed the effectiveness of the proposed algorithm. In addition, we also employed the GMMs in a particular OFD application, and the results of the online A/B test indicated the practical value of introducing a stochastic model for decision-making. This method can be extended to the modeling of distributions on other uncertain data. In future work, we will employ the current state and fine-grained features to build a predictive model, which can guide decision-making more accurately.

Acknowledgements This research is supported by the National Science Fund for Distinguished Young Scholars of China [No. 61525304], the National Natural Science Foundation of China [No. 61873328], and Tsinghua University—Meituan Joint Institute for Digital Life.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Janairo JIB (2021) Unsustainable plastic consumption associated with online food delivery services in the new normal. Clean Responsible Consumpt 2:100014
2. Meituan and China Hospitality Association (2020) Report on the development of Chinese take-out industry in 2019 and the first half of 2020. https://ncstatic.clewm.net/rsrc/2020/0628/09/84f7f3e18c6e2-7cb32227534f640bd45.pdf
3. Wu C-H, Ho J-M, Lee D-T (2004) Travel-time prediction with support vector regression. IEEE Trans Intel Transp 5(4):276–281
4. Zheng J, Wang S, Wang L, Chen J-F, Wang L, Hao J, Sun Z (2020) A two-stage algorithm for fuzzy online order dispatching problem. In: 2020 IEEE congress on evolutionary computation (CEC). https://doi.org/10.1109/CEC48606.2020.9185858
5. Zhou Y, Huang J, Shi J, Wang R, Huang K (2021) The electric vehicle routing problem with partial recharge and vehicle recycling. Complex Intell Syst 7:1445–1458
6. Wu H, Gao Y, Wang W, Zhang Z (2021) A hybrid ant colony algorithm based on multiple strategies for the vehicle routing problem with time windows. Compl Intell Syst. https://doi.org/10.1007/s40747-021-00401-1
7. Gutierrez A, Dieulle L, Labadie N, Velasco N (2018) A multi-population algorithm to solve the VRP with stochastic service and travel times. Comput Ind Eng 125:144–156
8. Zhan Y, Wang Z, Wan G (2021) Home service routing and appointment scheduling with stochastic service times. Eur J Oper Res 288(1):98–110

9. Kuo R, Wibowo B, Zulvia F (2016) Application of a fuzzy ant colony system to solve the dynamic vehicle routing problem with uncertain service time. Appl Math Model 40(23–24):9990–10001
10. Errico F, Desaulniers G, Gendreau M, Rei W, Rousseau L-M (2018) The vehicle routing problem with hard time windows and stochastic service times. EURO J Transp Logist 7(3):223–251
11. Binart S, Dejax P, Gendreau M, Semet F (2016) A 2-stage method for a field service routing problem with stochastic travel and service times. Comput Oper Res 65:64–75
12. Nasri M, Metrane A, Hafidi I, Jamali A (2020) A robust approach for solving a vehicle routing problem with time windows with uncertain service and travel times. Int J Ind Eng Comp 11(1):1–16
13. Tajik N, Tavakkoli-Moghaddam R, Vahdani B, Mousavi SM (2014) A robust optimization approach for pollution routing problem with pickup and delivery under uncertainty. J Manuf Syst 33(2):277–286
14. Bian B, Zhu N, Ling S, Ma S (2015) Bus service time estimation model for a curbside bus stop. Transp Res Part C Emerg Technol 57:103–121
15. Wang Y, Zheng Y, Xue Y (2014) Travel time estimation of a path using sparse trajectories. In: Proceedings of the 20th ACM SIGKDD, pp 25–34
16. Nguyen HD, Wang D, McLachlan GJ (2018) Randomized mixture models for probability density approximation and estimation. Inform Sci 467:135–148
17. Li W, Wang Z, Yuan Y, Guo L (2016) Particle filtering with applications in networked systems: a survey. Complex Intell Syst 2(4):293–315
18. Redner RA, Walker HF (1984) Mixture densities, maximum likelihood and the EM algorithm. SIAM Rev 26(2):195–239
19. Biernacki C, Celeux G, Govaert G (2003) Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. Comput Stat Data An 41(3–4):561–575
20. Martınez AM, Vitria J (2000) Learning mixture models using a genetic version of the EM algorithm. Pattern Recogn Lett 21(8):759–769
21. Kiranyaz S, Ince T, Yildirim A, Gabbouj M (2009) Fractional particle swarm optimization in multidimensional search space. IEEE Trans Syst Man Cybern B 40(2):298–319
22. Kwedlo W (2014) Estimation of parameters of Gaussian mixture models by a hybrid method combining a self-adaptive differential evolution with the EM algorithm. Adv Comput Sci Res 11:109–123
23. Biernacki C, Govaert G (1997) Using the classification likelihood to choose the number of clusters. Comput Sci Sta, 451–457.
24. Melnykov V, Melnykov I (2012) Initializing the EM algorithm in Gaussian mixture models with an unknown number of components. Comput Stat Data An 56(6):1381–1395
25. Pernkopf F, Bouchaffra D (2005) Genetic-based EM algorithm for learning Gaussian mixture models. IEEE Trans Pattern Anal 27(8):1344–1348
26. Pignat E, Calinon S (2019) Bayesian Gaussian mixture model for robotic policy imitation. IEEE Robot Autom Lett 4(4):4452–4458
27. Mirra J, Abdullah S (2021) Bayesian gaussian finite mixture model. J Phys Conf Ser 1725(1):012084
28. Liu G, Liu Y, Guo M, Li P, Li M (2019) Variational inference with Gaussian mixture model and householder flow. Neural Netw 109:43–55
29. Hershey JR, Olsen PA (2007) Approximating the Kullback Leibler divergence between Gaussian mixture models. In: 2007 IEEE international conference on acoustics, speech and signal processing-ICASSP, vol 4, pp 317–320
30. Chen L, Wang D, Gan Z, Liu J, Henao R, Carin L (2021) Wasserstein contrastive representation distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16296–16305
31. Vallender S (1974) Calculation of the Wasserstein distance between probability distributions on the line. Theor Prob Appl 18(4):784–786
32. Hauschild M, Pelikan M (2011) An introduction and survey of estimation of distribution algorithms. Swarm Evol Comput 1(3):111–128
33. Cheng R, He C, Jin Y, Yao X (2018) Model-based evolutionary algorithms: a short survey. Complex Intell Syst 4(4):283–292
34. Zheng J, Wang L, Wang J (2020) A cooperative coevolution algorithm for multi-objective fuzzy distributed hybrid flow shop. Knowl-Based Syst 194:105536
35. Shi W, Chen W-N, Gu T, Jin H, Zhang J (2020) Handling uncertainty in financial decision making: a clustering estimation of distribution algorithm with simplified simulation. IEEE Trans Emerg Top Comput Intel 5(1):42–56
36. Chandran BP, Selvakumar AI, Let GS, Sathiyan SP (2020) Optimal model parameter estimation of solar and fuel cells using improved estimation of distribution algorithm. Ain Shams Eng J 12(2):1693–1700
37. Blei DM, Griffiths TL, Jordan MI (2010) The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. J ACM 57(2):1–30
38. Ferguson TS (1973) A Bayesian analysis of some nonparametric problems. The Ann Stat 1(2):209–230
39. Pfeifer T, Protzel P (2019) Expectation-maximization for adaptive mixture models in graph optimization. In: 2019 international conference on robotics and automation (ICRA), pp 3151–3157.
40. Sinaga KP, Yang M-S (2020) Unsupervised K-means clustering algorithm. IEEE. Access 8:80716–80727
41. Chen J, Wang S, Wang L, Zheng J, Cha Y, Hao J, He R, Sun Z (2020) A hybrid differential evolution algorithm for the online meal delivery problem. In: IEEE congress on evolutionary computation (CEC). https://doi.org/10.1109/CEC48606.2020.9185792
42. Chen J, Wang L, Wang S, Wang X, Ren H (2021) An effective matching algorithm with adaptive tie-breaking strategy for online food delivery problem. Compl Intell Syst. https://doi.org/10.1007/s40747-021-00340-x