基于演化博弈及微观仿真的动态定价模型

摘要

本文针对拍照定价问题，提出了基于动态演化博弈、微观仿真、多目标优化等模型的定价方案。逐步优化了任务定价策略。

针对问题一，提出了任务密度、会员密度以及任务的地理位置三个定价影响因素，通过**Spearman相关分析**筛选出任务密度和地理位置两个因素。然后利用**多元逐步回归分析定价**与其所筛选因素之间的关系，得到了定价规律函数。对不同属性的任务进行**Q型聚类分析**，通过对任务进行横向与纵向的比较分析，得出任务未完成与供求关系、地理位置、价格有关。

针对问题二，通过分析用户方和 APP 方的利益博弈关系，建立了**动态演化博弈的定价模型**，为达到双方利益最大化的平衡状态，令二者收益期望相同，代入已知数据，得到定价方案二。并建立**BP神经网络预测模型**，利用附件一的部分数据进行训练学习，剩余数据进行检验。预测出新定价方案下任务的完成情况，得出在价格稳定的前提下，该定价方将任务的完成率提高了12.60%。

针对问题三，在博弈定价模型的基础上，建立**微观仿真模型**模拟用户完成打包任务及单独任务的情况。引入并仿真计算出**任务效率**K，得到将任务打包发布的定价方案三。求解出任务打包发布使任务完成率提高了6.12%，价格减少了3.1%。由此得到了任务的基础价格G_b，又考虑到区域之间的不平衡度，提出了不同区域的**价格补偿值**ΔG_i，建立了**多目标优化模型**求解。最后得到了任务i的总定价为$G_i = G_b + \Delta G_i$。相对于方案三任务完成率提高了5.29%，并减少了各个区域的不平衡度。

针对问题四，对打包方式做进一步讨论，利用$\mathbf{C-means}$**聚类**确定出最优的打包中心点，并结合活动半径对中心点周围的任务进行打包，由此改进所有的定价方案。再综合比较不同地区和不同定价方案的完成效果，发现本文的三个定价方案逐渐地提升了任务完成度，降低了个地区的不平衡度。

最后对模型进行了优缺点分析和推广。

关键词：动态演化博弈;BP神经网络预测;微观仿真;多目标优化;回归分析

一、问题的重述

1.1 背景资料与条件

“拍照赚钱”是移动互联网下的一种自助式服务模式。用户下载 APP，注册成为 APP 的会员，从 APP 上领取需要拍照的任务（比如上超市去检查某种商品的上架情况），赚取 APP 对任务所标定的酬金。这种基于移动互联网的自助式众包平台，相比传统市场调查方式大大节省调查成本，而且有效保证了数据真实性，缩短了调查的周期。因此 APP 成为该平台运行的核心，而 APP 中的任务定价又是其核心要素。若定价不合理，有的任务就会无人问津，导致商品检查的失败。

1.2 需要解决的问题

附件一是一个已结束项目的任务数据，包含每个任务的位置、定价和完成情况；附件二是会员信息数据，包含了会员的位置、信誉值、参考其信誉给出的任务开始预订时间和预订限额，原则上会员信誉越高，越优先开始挑选任务，其配额越大（任务分配时实际上是根据预订限额所占比例进行配发）；附件三是一个新的检查项目任务数据，只有任务的位置信息。需解决以下问题：

1、研究附件一中项目的任务定价规律，分析任务未完成的原因。

2、为附件一中的项目设计新的任务定价方案，并和原方案进行比较。

3、实际情况下，多个任务可能因为位置比较集中，导致用户会争相选择，一种考虑是将这些任务联合在一起打包发布。在这种考虑下，修改前面的定价模型，并分析对最终的任务完成情况的影响。

4、对附件三中的新项目给出新的任务定价方案，并评价该方案的实施效果。

二、问题的分析

2.1 问题一的分析

问题一要求研究定价规律，并分析任务未完成的原因。选择任务密度、会员密度以及任务的地理位置三个影响因素，本着完备性、代表性、目的性、可比性的原则，选择出对定价影响较大的因素并建立函数关系。对所有的任务进行分类，结合函数关系，通过对任务的横向和纵向对比，总结出任务未完成的原因。

2.2 问题二的分析

问题二要求设计新的定价方案，并与原方案对比。引入供求平衡补偿值与任务难度两个因素，与问题一中的三个因素共同作用于任务完成情况。由于用户希望价格尽可能高，而 APP 平台希望价格较低，二者构成博弈关系。当两方的净

收益相同时，达到利益最大化，此时的任务价格为最佳。并根据最佳定价及其他因素预测出任务的完成情况，与原结果对比。

2.3 问题三的分析

问题三要求在将距离较近的任务联合发布的前提下，制定新的定价方案。引入任务吸引程度的概念区分打包任务与单独任务。选取部分城市道路，采用随机投放打包及单独任务的形式对任务分配情况进行分析，从而制定出更合理的定价方案。并将不同区域的经济、交通水平纳入考虑范围内，对该价格方案进行优化。

2.4 问题四的分析

问题四要求给定新的定价方案，并进行评价。考虑结合前三问的任务定价模型，依次从会员与 APP 平台收益的博弈关系、打包任务与单独任务吸引能力的比较、区域之间的发展平衡三个方面对定价方案进行优化调整。将不同方案间的差异及不同城市的差异纳入考虑范围内，考察实施效果。

三、模型的假设

(1)假设不考虑该 APP 与其他 APP 的竞争关系。
(2)假设当会员用户接受某个任务或打包任务后，完成之前不再接受其他任务。
(3)假设会员用户在完成任务时，选择距离最短的道路和最便捷的交通工具。
(4)假设在分析任务打包前后的完成情况时，会员移动的速度是相同的。

四、符号的说明

符号	说明
$Longtitude$	任务的经度
$Latitude$	任务的纬度
d_0	任务的活动半径
ρ_{i1}	任务 i 所在的活动区域的任务密度
ρ_{i2}	任务 i 所在的活动区域的用户密度
$price_i$	任务 i 的价格

C_i	任务 i 与市中心的距离
P_i	任务 i 的供求比率
D_i	任务难度
Q_i	任务完成率
G_b	基础价格
ΔG_i	价格补偿值

注：其它符号将在下文中给出具体说明。

五、模型的建立与求解

5.1 问题一：多元线性逐步回归模型

5.1.1 模型准备：数据处理

根据已给数据，对不同地点的任务、会员进行分析，以便探究价格制定规律。我们对所给数据作如下几步处理。

1、绘制二维散点图

由附件一所给数据，制定任务完成情况的二维散点图，观察任务完成情况：

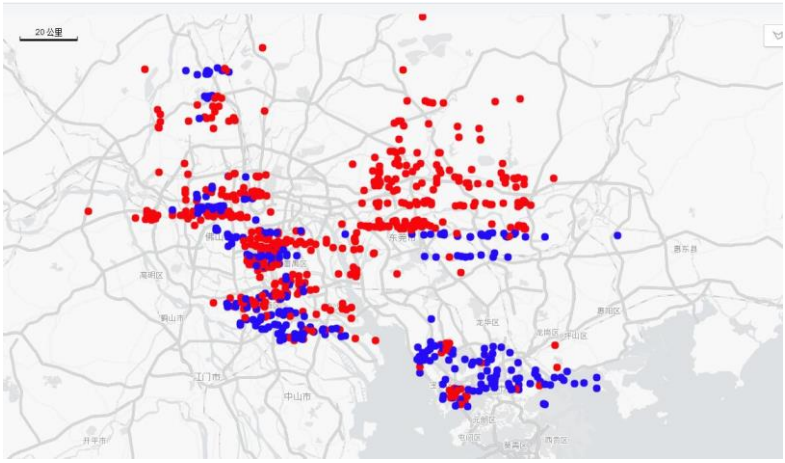

图 1 任务的分布图

我们发现未完成任务的分布均在佛山市及东莞市南部地区附近。着重分析该集中区域的价格规律，便可找出任务未完成的原因。

2、建立任务间的距离矩阵

为简化问题，将任务所在地点的经纬度数据转化为两点间距离矩阵。通过附

件一中任务的经度 *Longtitude*、纬度 *Latitude*，将任务 M 与 N 的经纬度坐标简化为 (LoM, LaM)、(LoN, LaN)。经查阅资料[1]，得到任务 M 与 N 之间的距离 A：

$$\begin{cases} A = R * \text{Arccos}(C) * \pi / 180 \\ C = \sin(La'M) \cdot \sin(La'N) \cos(LoM - LoN) + \cos(La'M) \cos(La'N) \\ La'M = 90 - LaM \end{cases} \quad (1)$$

式中，R 为地球半径，取 6356.755 千米。

通过公式，得到任务间的地表距离，将其简化为一个 835*835 的矩阵 $A_{(835\times 835)}$：

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1(835)} \\ A_{21} & A_{22} & \cdots & A_{2(835)} \\ \cdots & & \cdots & \cdots \\ A_{(835)1} & & \cdots & A_{835(835)} \end{pmatrix} \quad (2)$$

A_{ij} 代表任务 i 到 j 的地表距离。

3、建立会员与任务间的距离矩阵

会员与任务之间的距离是在接受任务时需要考虑的因素之一。同建立任务间的距离矩阵一样，建立 1877 个会员所在的位置与 835 个任务所在位置的地表距离矩阵 $B_{(835\times 1877)}$。B_{ij} 代表任务 i 到会员 j 的地表距离。

5.1.2 影响因素的选取

1、活动半径的概念

我们认为，受交通、经济等多种外界因素的影响，用户仅在一定的区域内活动。也就是说，只有任务周围的用户才会接受该任务。以任务为圆心，接受任务的活动半径为 d_0。当用户落入该区域内，则认为用户会接受该任务，否则不接受。

由附件一得知任务主要分布在广州、深圳、佛山、东莞四座城市。根据经验和查阅资料，城市的面积越大、交通越便利、人口越少，且周围的生活设施越不充足时，居民活动范围越大，接受任务的会员分布越广。得到任务活动半径 d_0 为

$$d_0 = \sqrt{\frac{s^2 e^{-\lambda} k}{a\pi}} \quad (3)$$

式中：s 为四座城市总面积；k 为城市交通水平，经查阅资料，取 $k = 0.8$；λ 为四座城市的人均 GDP，a 为四座城市总人口。

城市各项指标数据见附录一，最终求解得到任务的活动半径 d_0=1.5299 千米。

2、影响因素

利用活动半径 d_0，结合以下影响因素来表征任务被接受的可能性，并反映到

任务的定价规律当中。

◆**任务密度** ρ_{i1}

周围任务密度表征在一个任务的活动范围之内所有任务的数目。定义任务 i 所在的活动区域 $(R < d_0)$ 的任务密度 ρ_{i1} 为

$$\begin{cases} \rho_{i1}=n_1+1 \\ s.t.\ A_{ij} \le d_0 \end{cases} \tag{4}$$

式中: n_1 为活动半径范围内周围任务的数目。由于只有在任务活动半径内的用户才会接受该任务，当任务密度较高时需要适当提高价格，保证任务的完成度。

◆**用户密度** ρ_{i2}

用户密度指的是在任务活动半径范围内该 APP 会员数目的平均数量。定义任务 i 所在的活动区域 $(R < d_0)$ 的用户密度 ρ_{i2} 为

$$\begin{cases} \rho_{i2}=n_2 \\ s.t.\ B_{ij} \le d_0 \end{cases} \tag{5}$$

式中: n_2 为活动半径范围内会员的数目。当用户密度较高时，表征该区域愿意接受任务的人数越多，可以适当降低任务定价。

◆**任务地理位置** C_i

由任务完成情况的二维散点图，发现未完成任务的分布较集中。说明任务的完成情况与地理位置具有相关性。定义广州、深圳、佛山、东莞四座城市中心位置坐标 $X_i(i=1,2,3,4)$ 分别为：

表 1 四座城市中心位置坐标

	广州	深圳	佛山	东莞
X_i	(23.16,113.23)	(22.62,114.07)	(23.05,113.11)	(23.04,113.75)

利用公式（1）计算任务 i 与四个市中心间的距离，记为 $d_{i1}, d_{i2}, d_{i3}, d_{i4}$，从中选取最小值 C_i，衡量该任务位置的偏僻程度：

$$C_i = \min(d_{i1}, d_{i2}, d_{i3}, d_{i4}) \tag{6}$$

最终得到了 835 个已完成任务的地理位置矩阵 $C_{835\times1}$。

当任务所在的位置距离市中心越近时，代表该位置的经济状况较好，交通条件便利，可适当提高任务定价。

5.1.3 模型的建立与求解

1）影响因素的相关性分析

如图 2 所示，我们初步建立了三个影响定价规律的因素，并分别进行量化。

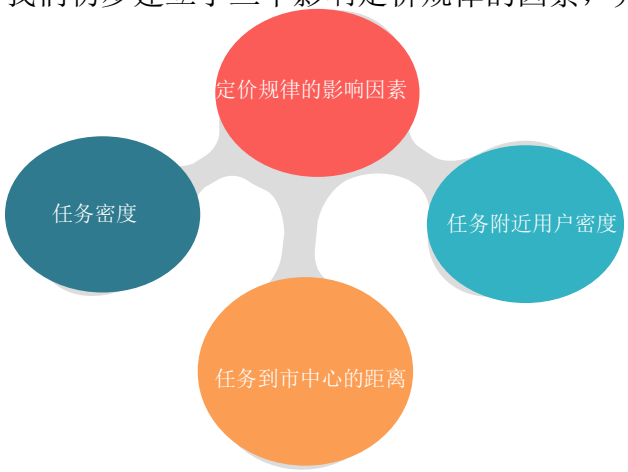

图 2 定价的三个影响因素

以上三个影响指标均与 835 个任务所在位置相关，为避免各影响因素之间存在关联性，分别量化出三个影响因素对价格的影响程度。

采用 *Spearman* 秩相关检验的方法，以任务附近用户密度与价格的相关度为例，ρ_{i2} 和价格 $price_i$ 分别按照从大到小的顺序排序，所在位置的秩分别记作 R_i，S_i [2],构成 835 对秩 $(R_1,S_1)、(R_2,S_2)\cdots(R_{835},S_{835})$。每一对秩次差为$d_i=R_i-S_i$。则

$$r_s=\frac{\sum_{i=1}^{n}(R_i-\overline{R})(S_i-\overline{S})}{\sqrt{\sum_{i=1}^{n}(R_i-\overline{R})^2\sum_{i=1}^{n}(S_i-\overline{S})^2}}=1-\frac{6\sum_{i=1}^{n}d_i^2}{n(n^2-1)} \tag{7}$$

式中：r_s 为 *Spearman* 相关系数；$\overline{R}=\frac{1}{n}\sum_{i=1}^{835}R_i$，$\overline{S}=\frac{1}{n}\sum_{i=1}^{835}S_i$。

r_s 的范围是 $(-1,1)$，$|r_s|$ 越接近 1，二者间的相关度越高。

2）相关性结果分析

根据以上模型，求解出三个影响因素对定价的影响程度如下(带**表明通过了显著性为 0.01 的检验)：

表 2 影响因素与价格之间的相关系数

	任务密度	人员密度	与市中心的距离
相关系数	-0.646**	-0.042	0.403**
显著性	0.000	0.680	0.000

如表 2 所示，任务密度、任务与市中心的距离与价格之间的相关系数的绝对值较大，可以在0.01的显著性水平下认为这两个因素与价格存在着较强的相关关系。并得到以下结论：

任务密度与价格之间存在着负相关关系，任务与市中心的距离和价格之间为较强的正相关关系。而人员密度对价格的影响很小。说明该平台在人员密度相对较小的地方投入的任务可能本来就较少，或者这些地方的交通可能较为便利，完成任务较容易。因此探究任务的价格规律时排除人员密度因素。

3）多元线性逐步回归

采取逐步回归[3]的方法，将影响价格的因素按照影响程度由大到小的顺序进行线性回归。以任务密度 ρ_{i1} 为例：

Step1：求出 835 个任务所在地的任务密度 ρ_{i1} 的平均值 $\overline{\rho_{i1}}$ 和 835 个价格的平均值 $\overline{price_i}$，以及离差平方和 L：

$$\begin{cases} L\rho_{i1} = (\rho_{i1} - \overline{\rho_{i1}})^2 \\ Lprice_i = (price_i - \overline{price_i})^2 \end{cases} \tag{8}$$

对数据标准化处理：

$$\rho_{i1s} = \frac{\rho_{i1} - \overline{\rho_{i1}}}{\sqrt{L\rho_{i1}}}, price_{is} = \frac{price_i - \overline{price_i}}{\sqrt{Lprice_i}} \quad i = 1, 2, \cdots p \tag{9}$$

Step2：求出 835 个任务所在地的任务密度 ρ_{i1} 和 835 个价格 $price_i$ 的相关系数矩阵。使用*Pearson* 相关系数进行计算：

$$\gamma = \frac{N\sum \rho_{i1} price_i - \sum \rho_{i1} \sum price_i}{\sqrt{N\sum \rho_{i1}^2 - (\sum \rho_{i1})^2}\sqrt{N\sum price_i^2 - (\sum price_i)^2}} \tag{10}$$

根据该系数计算出任务密度与价格之间的相关系数矩阵。

*Pearson*相关系数[4]的检验统计量为

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} \tag{11}$$

式中：t 满足$n-2$个自由度的t 分布。

Step3：假设对第$k(k=1,2)$个影响因素进行回归，相关系数矩阵转化为

$$R^{(k)} = (r_{ij}^{(k)}) \tag{12}$$

将标准化后值的偏回归平方和记为

$$V_{ij}(k)=\frac{(r^{k}_{ij,(p+1)})^2}{r^{k}_{ij,ij}} \tag{13}$$

选取偏回归平方和中的最大值，在给定显著性水平 α 后，作显著性的 F 检验。若未能通过显著性检验，就排除该因素。在对任务密度 ρ_{i1} 进行计算后，引入任务与市中心的距离 C_i 作相同的计算。最终得到回归方程为

$$\frac{price_i-\overline{price_i}}{\sqrt{Lprice_i}}=r^{k}_{ij,(p+1)}\frac{\rho_{i1}-\overline{\rho_{i1}}}{\sqrt{L\rho_{i1}}}+r^{k}_{ij,(p+1)}\frac{C_i-\overline{C_i}}{\sqrt{LC_i}} \tag{14}$$

4）回归模型的结果及分析

根据以上模型，得到的各参数的结果见附录二。得到回归方程为：

$$price_i=0.035\rho_{i1}-0.399C_i+71.736 \tag{15}$$

根据以上公式，可以看出，任务的地理位置对价格的负相关影响较大，密度对价格的正相关影响较小。在分析任务未完成原因时，需要综合考虑这两个因素。

5.1.4 任务未完成原因分析

1）对任务进行 Q 型聚类分析

为进一步分析任务未被完成的原因，根据任务的三种属性：任务密度 ρ_{i1}、用户密度 ρ_{i2}、任务与市中心的距离 C_i，对所有任务进行 Q 型聚类。

首先对量化后任务的各个因素的数值进行标准化处理。以任务密度 ρ_{i1} 为例：

对于 835 个任务密度而言，标准化后的值为

$$\rho_{i1}^{*}=\frac{\rho_{i1}-\overline{\rho_{i1}}}{s_{\rho i1}}(i=1,2,\cdots 835) \tag{16}$$

式中：$\overline{\rho_{i1}}$ 为 835 个任务密度的平均值，$s_{\rho i1}$ 为 835 个任务密度的方差。

令 x_{ij} 表示第 i 个任务的第 j 个影响因素，d_{ij} 表示第 i 个任务与第 j 个任务之间的欧式距离，为

$$d_{ij}=\left[\sum_{k=1}^{p}\left(x_{ik}-x_{jk}\right)^2\right]^{1/2} \tag{17}$$

首先将 835 个任务中的每个任务看作一类，类间的距离与任务间距相等。采取最短距离法[5]，将距离最小的一对任务合并为一类，并计算出与其他类的距离，再次将距离较小的进行合并，直到所有的任务被合并为一类为止。最终完成分类。

2）Q 型聚类结果及分析

供求比率可以衡量任务与用户的匹配程度。利用任务密度 ρ_{i1} 表征任务的供给量，用户密度 ρ_{i2} 表征需求量，二者比值即为供求比率 P_i。当一定范围内供给量大于需求量时，意味着任务的数目出现了饱和，未完成的几率较高。而供给量小于需求量则说明该范围内会员数目较多，任务的完成率会较高。供给量与需求量相等时的情况介于二者之间。

根据上述 Q 型聚类的方法，得到以下四种类型的任务，如图 3 所示：

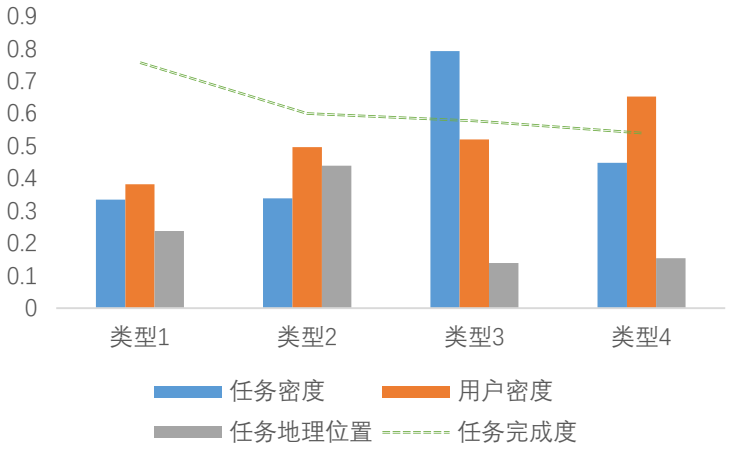

图 3 聚类分析结果

① 类型一（供求均衡类）中，供求比率几乎为 1，任务密度和用户密度相对平衡，任务地点距离市中心较近，交通较便利。因此该类型的任务完成率是最高的，达到了 76%。

② 类型二（位置偏僻类）中，任务的供求比率小于类型一，可接受任务的人数相对较多，然而任务的完成率却不如类型一。经分析知该类型任务距离市中心是最远的，导致完成任务的难度也较大。从侧面说明任务距离市中心的距离对任务完成程度的影响更大一些。

③ 类型三（密度集中类）中的任务密度是最大的，与市中心的距离也较近，该地区交通便利。但此时用户密度与任务密度相差较多，供求比率高，因此任务完成率相对于前两种情况较低一些。

④ 类型四（价格过高类）中，用户密度的数量最多，供求比率较低，距离市中心也较近，但此时的任务完成度却最低。经分析数据可知，该地区任务的价格相对较低，因此导致完成率较低，为 54.01%。

综上所述，任务未完成的原因是多方面的，需要从供求关系、任务的地理位置、任务价格的设置等多方面分析。

3）Q 型聚类结果的验证

为了进一步验证聚类结果，得到结果如图 4 所示:

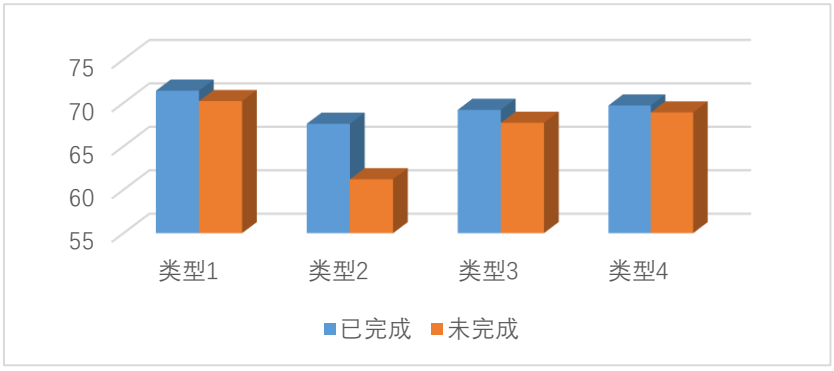

图 4 已完成任务和未完成任务的平均价格

经过对比，发现已完成任务的价格普遍高于未完成任务。说明在其他条件不变的情况下，价格较低会导致任务的完成率较低。

5.2 问题二：动态演化博弈及神经网络预测模型

5.2.1 模型准备

将原数据中的价格方案记作方案一。为制定新的定价方案，引入两个指标：

1）供求比率 P_i

若任务密度为 ρ_{i1}，用户密度为 ρ_{i2}，则任务 i 的供求比率 P_i 为

$$P_i = \rho_{i1}/\rho_{i2} \tag{18}$$

供求比率较大时，说明该区域内任务密度饱和，任务的完成率可能较低。供求比率较小时与之相反。特别地，当 P_i=1时，说明任务与用户的数量达到了平衡。

2）任务难度 D_i

由实际可知，任务困难程度是任务 i 到会员 j 的距离 B_{ij}、任务与市中心距离 d_0 的增函数，当用户离任务越远，且任务所在地交通条件较差、经济水平较低时，完成任务困难程度越大。由聚类分析可知，当任务的地理位置相近时，价格越高的任务往往相对困难一些。建立任务困难程度的函数关系，并用常数 k 进行修正。

$$D_i = kB_{ij}e^{-d_0}price_i^{1.1} \tag{19}$$

5.2.1 模型的建立与求解

1）动态演化博弈模型

会员在接受任务时主要关注任务的价格、难度，而 APP 平台关注任务的价格与完成度两方面。会员希望在价格尽可能高，而 APP 平台希望控制价格在较低的范围内。二者的期望构成博弈关系。

①建立博弈矩阵

因而建立动态的演化博弈模型[6]，使二者的收益的期望值达到相同的状态，此时便认为价格是合理的。

假定 APP 平台和用户的状态策略分别为S_1、S_2，这两个集合中的任务均有接受与否两种状态，即

$$\begin{cases} S_1=\{\text{任务}i\text{被接受}(S_{11})，\text{任务}i\text{未被接受}(S_{12})\} \\ S_2=\{\text{接受任务}i(S_{21})，\text{未接受任务}i\ (S_{22})\} \end{cases} \tag{20}$$

经分析可知，APP 平台的状态策略为任务i被接受时，获得的收益由企业的支付资金F_1衡量，减去支付给用户的价格$price_i$，便得到了在任务被接受时的净收益$a_1=F_1-price_i$;当任务i未被接受时，APP 平台需要支付给企业F_2的违约金，净收益为$a_2=-F_2$。

当会员接受任务i时，除去交通、时间成本的量化值D外，获得的收益为 APP 平台所制定的价格$price_i$，即净收入为$b_1=price_i-D$；当未接受任务时，不会获得收益，令净收益$b_2=0$。

综上所述，将上述指标进行两两配对，建立表 3 的博弈矩阵：

表 3 博弈矩阵的建立

	S_{21}	S_{22}
S_{11}	a_1,b_1	a_2,b_2
S_{12}	a_2,b_2	a_2,b_2

②动态博弈模型构建

构建博弈矩阵后，假定 APP 平台中被接受，即S_{11}的概率为p。用户接受任务i，即S_{21}的概率为q，分别可以得到不同情况下 APP 平台和用户的净收益期望的估计值[7]。

- 从 APP 方的角度，任务i定价被接受的概率，即S_{11}的概率p，取决于供求相对不平衡度以及任务相对难度，且均呈负相关关系，因此$p=\left|1-P_i\right|D_i/D_{\max}$。
- 由经济学中的需求 logisitic 函数，从用户的角度，拒绝任务i定价的概率取决于实际报价与期望价格E_a的差异，差异越大，拒绝的概率越大。期望的价格E_a服从价格上下限的均匀分布，因此用户接受任务i的概率为$q=e^{-(E_a-price_i)/E_a}$。

对于 APP 平台而言，当任务i被接受时，获得净收益的期望为

$$E_{11}=qa_1+(1-q)a_2 \tag{21}$$

同理，当任务i不被接受时，期望值为$E_{12}=qa_2+(1-q)a_2=a_2$。得到 APP 平台总的净收益的值为

$$E_1 = pE_{11} + (1-p)E_{12} = pqa_1 + (1-pq)a_2 \tag{22}$$

对于用户而言，用相同的方法，当接受任务 i 时，获得的净收益的期望值为 $E_{21}=pb_1$。当不接受任务 i 时，期望为 $E_{22}=0$。会员用户总的净收益的值为

$$E_2 = qE_{21} + (1-q)E_{22} = pqb_1 \tag{23}$$

③模型的演化过程

经查阅资料，APP 平台任务被接受的概率随时间不断变化，变化率[7]为

$$\frac{dp}{dt} = pq(1-p)(a_1 - a_2) \tag{24}$$

用户接受任务 i 的概率为

$$\frac{dq}{dt} = b_1 pq(1-q) \tag{25}$$

当 $\frac{dp}{dt}$=0时，任务 i 定价被接受的概率相对稳定；$\frac{dq}{dt}$=0 时，用户接受任务 i 的概率相对稳定。演化过程如图 5：

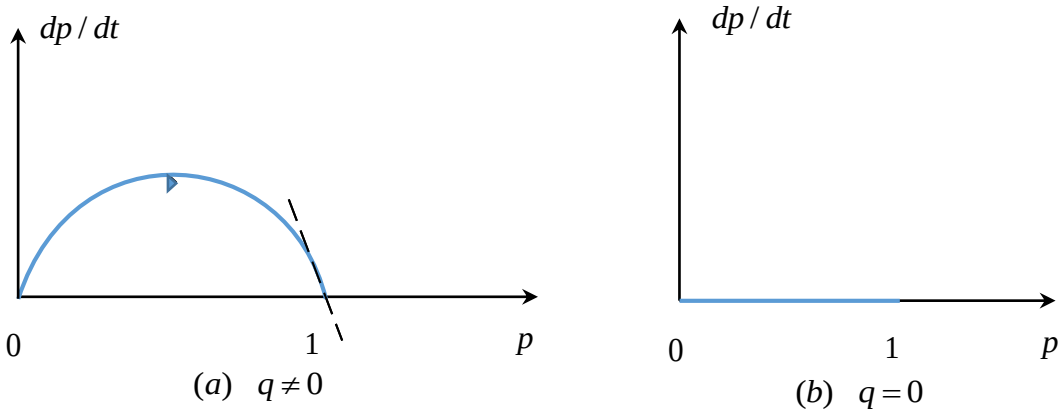

图 5 模型的演化过程

④建立新的定价方案

当 APP 平台中的任务净收益期望值与用户收益的期望值相等时，认为达到了平衡状态，此时价格较合理，使 APP 平台与用户利益最大化。令 $E_1 = E_2$，消除公共参数，并代入已知的概率值，得到 835 个任务的新定价方案，记作方案二。

2）利用 BP 神经网络预测模型进行对比验证

神经网络作为由神经元构成的复杂网络系统，由三层构成。可以仅仅通过附件一中给定的任务位置、价格数据，得到原定价方案中各影响因素到任务价格的映射规律，方便与新定价方案对比验证。

将 BP 神经网络的输入层设置为与价格相关的 5 个影响因素。输出层为任务的完成情况，已完成取 1，未完成为 0。中间的隐含层个数根据经验公式 $(2k+1)$ 依次增加的方法进行计算。其中，k 为输入层的个数。如图 6 所示：

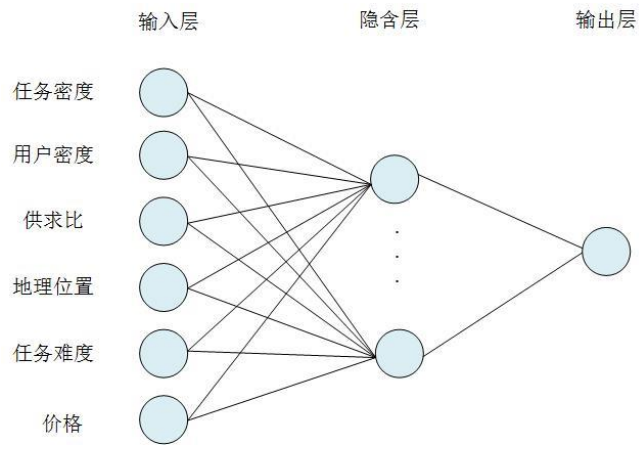

图 6 神经网络的基本结构

利用神经网络建立预测模型的具体过程如下：

Step1: 归一化输入因素数值。

随机在 835 个任务样本中选取 400 个，将输入层中 5 个影响因素的具体数据归一化，并输入到已设置的网络中。

Step2: 计算 400 个任务的输出层数值

对 400 个任务进行迭代，计算输出层的值，观察任务完成度与实际是否相符。

Step3: 若输出的任务完成情况与原定价中的相同，则本次训练结束；否则继续进行训练，达到规定的数值后，训练完成。

经过 1000 次训练，达到设定的收敛精度10^{-4}。在剩下的任务中随机抽取 100 个，将各因素数值输入到网络中，对该网络进行验证。发现结果与原数据基本符合，则认为该模型准确。检验结果如下：

表 4 神经网络的检验结果

	检验个数	正确个数	正确率
未完成	43	42	97.67%
完成	57	52	91.23%
总体	100	94	94%

在得到能够准确反映各因素与任务完成情况的预测模型后，将新的定价方案中任务的地理位置、供求概率、新的价格中的信息作为输入层，代入神经网络预测模型，输出了 835 个新方案的任务完成情况，与原方案进行对比。

5.2.3 结果分析

1、价格对比

根据动态博弈模型得到新的定价方案二。随机选取 30 组未完成的任务，比

较原方案与方案二的价格，如图 7 所示：

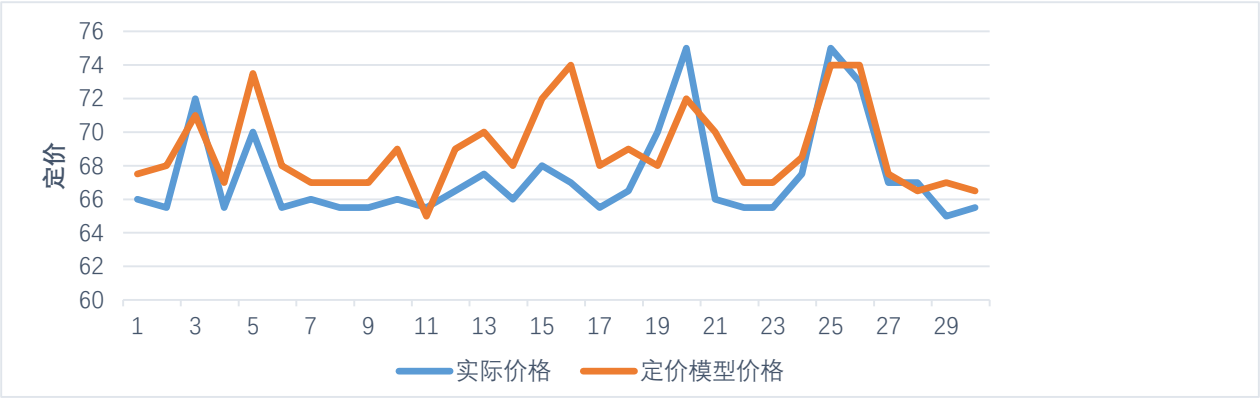

图 7 定价模型在失败任务数据集上的定价结果

由图可知，建立 APP 平台和用户的博弈模型后，确定了新的定价方案，使双方的利益共同达到最大化，此时未完成任务的价格普遍提升。经分析可知，在任务价格较低时，用户的净收益的期望值也较低，因此导致用户拒绝接受该任务的可能性较大。在调整定价方案为方案二后，用户的利益期望值得到提升。

2、任务完成率对比

将新的定价方案中任务的地理位置、供求概率、新的价格中的信息作为输入层，代入神经网络预测模型，得到任务的完成情况，用每一种类型的任务完成概率来表征，如图 8 所示。

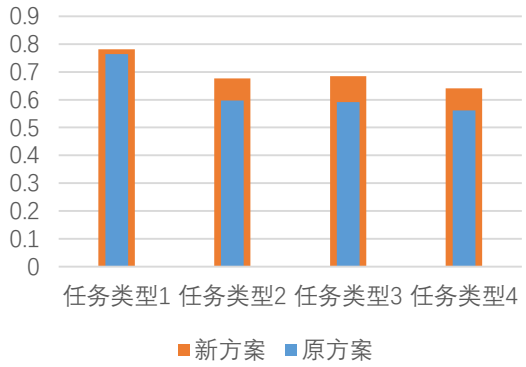

图 8 定价方案改变前后任务的完成率

经过分析可知，在设计新的评价方案后，APP 平台中的任务净收益期望值与用户收益的期望值相等，二者的利益达到最大化，由于任务未完成的原因与价格设置的不合理相关。在适度调整定价后，此时不同类型用户接受任务的概率普遍得到提升。进一步验证了新的定价方案的合理性。

5.3 问题三：微观动态仿真模型和多目标优化模型

为分析打包发布的任务分配情况，建立模拟仿真模型对打包前后的任务分配

情况进行模拟仿真，建立微观的动态仿真模型。

5.3.1 微观动态仿真模型

为分析打包发布的任务分配情况，建立模拟仿真模型对打包前后的任务分配情况进行模拟仿真，建立动态的仿真模型。

1、模型准备

定义平均效率为单位时间完成的任务量，假设单独任务和打包任务的效率分别为 k_1， k_2，价格分别为 M_{i1}， M_{i2}。则在单位时间内完成单独和打包任务的收益分别为 k_1M_{i1}、k_2M_{i2}，收益越高，对用户的吸引程度越高。

APP 平台与用户在任务价格上具备博弈关系，当 $k_1M_{i1}=k_2M_{i2}$ 时，两种任务对用户的吸引程度是相同的。在这种情况下，根据用户的信誉值可以对打包后的任务进行合理分配，并产生新的定价方案 M_{i2}。

为简化问题，我们假设：

1)不考虑会员用户移动速度的差别。

2)完成任务的过程中，会员用户总倾向于选择距离最短的路径。

3)当某会员用户接受某个任务或打包任务后，完成之前不再接受其他任务。

4)当一个任务同时落入多个会员用户的活动半径中时，按信誉值大小分配。

2、模型的建立与求解

根据假设，建立仿真模型，对打包前后的任务接受、完成情况进行模拟对比。

1）关于打包与未打包任务动态仿真系统初始化

在地图上截取北纬 20° 到 35°，东经 110° 到 125° 的正方形区域，并在地图上随机投放5个中心任务点，用欧几里得距离衡量任务间距：

$$d_{ij}=\sqrt{\left(x_i-x_j\right)^2+\left(y_i-y_j\right)^2} \tag{26}$$

选择与每个中心点距离500米内的区域，随机投放不同数目的任务数。将中心点与投放的任务数打包，形成5组独立的打包任务。并在地图上随机投放100个单独任务。同时将150个会员用户随机置入道路中。初始化如图 9 所示：

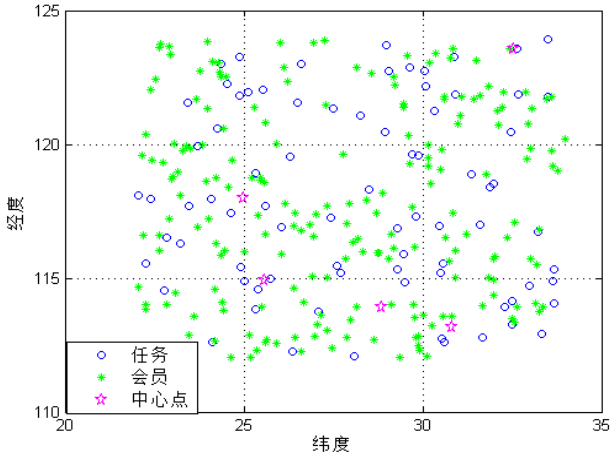

图 9 初始时刻的仿真情况

2）模拟仿真的具体分析

①会员完成任务的仿真

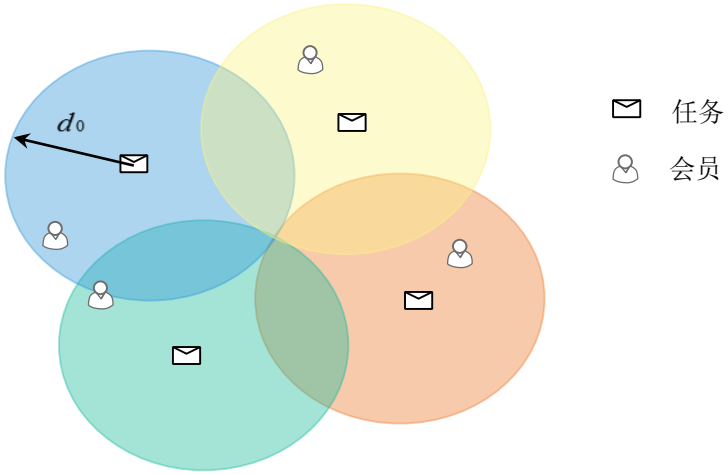

图 10 任务与会员位置的仿真情况

对该仿真进行以下分析：

• **活动半径：**在第一问中，我们已知只有任务周围的用户才会接受该任务。以任务为圆心，引入接受任务的活动半径 d_0。当用户落入该区域内，则认为用户有可能接受该任务，否则不会接受。任务活动半径为 d_0=1.5299 米。

• **会员用户运动规律**:在随机将 100 个打包任务和 100 个单独任务置入仿真系统后，我们认为任务是固定不动的，会员用户在接受任务之前以每秒钟 1.5 米/秒的速度随机运动，遇到路口时前行的概率是 50%，右转为 30%，左转为 20%。

• **会员接受及完成任务：**由活动半径的定义，在会员用户的随机移动过程中，若与某种任务的距离小于任务活动半径时，可以开始接受任务。

若同一任务具有多个会员有接受意向时，选择信誉值最高的会员用户接受任务。在抵达任务所在地过程中，排除绕远路的情况，当最近道路有多种选择方式

时，遇到路口时前行的概率是 50%，右转为 30%，左转为 20%。

②实现仿真的过程

根据以上分析，对用户完成任务的情况进行模拟。取迭代周期为 1 秒，用户在每个周期内移动1.5米。每一次迭代后返回以下信息：

- 判断会员用户与任务的距离是否小于1.5299米，若会员用户不在任何任务的活动范围内，则随机移动。
- 若用户距离某个任务小于活动半径时，需要判断在此范围内是否具有其他用户。若多个用户存在着竞争关系，优先选择信誉值大的用户完成任务，同时对用户的任务限额进行判断，若用户的任务限额已为0，则选择信誉值在其之后的用户。
- 若已确定用户完成某个任务，将该用户的任务限额减一。选择距离最短的路径抵达任务所在地。当最近道路有多种选择方式时，遇到路口前行概率是 50%，右转为 30%，左转为 20%。

3、动态仿真模型结果及分析

基于上述仿真模型，模拟出打包任务和单独任务的分配完成情况。设定迭代时间为 3600 秒，统计出这一个小时中打包任务的完成量 N_1 和单独任务的完成量 N_2 曲线如图 11 所示：

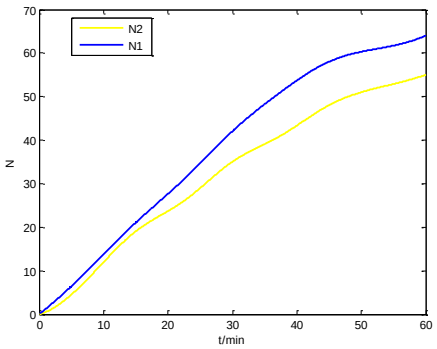

图 11 任务完成量随时间的变化。N_1:打包任务；N_2：单独任务

如图 11 所示，随着时间的推移，打包任务和单独任务的完成量都在不断增加，最终均达到了 50%以上。但打包任务的完成情况始终比单独任务要好，说明将任务打包可以提高任务的完成率。紧接着，求解出单位时间内单独任务和打包任务完成的任务量 k_1 和 k_2。由分析可知，当单独任务与打包任务对用户的吸引力相同，即 $k_1M_{i1}=k_2M_{i2}$ 时，可求解出打包后的任务价格 M_{i2}。

将方案二中的835个价格方案作为 M_{i1} 代入以上公式，最终求解出835个任务打包价格方案 M_{i2}，记作方案三。并与方案二的价格进行对比。如图 12 所示：

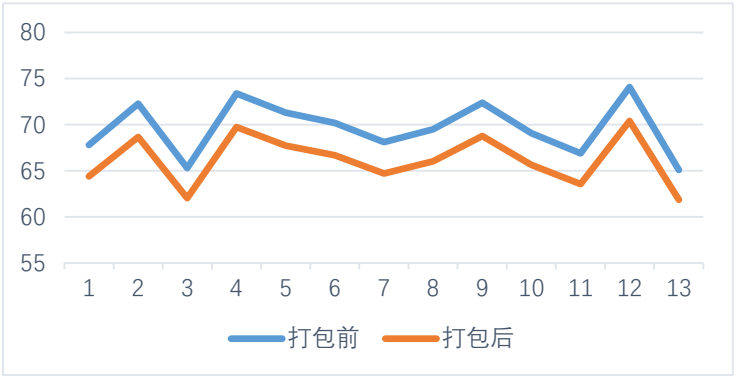

图 12 任务打包前后的价格

经过分析可知，将任务打包后再发布，与任务单独发布相比，虽然任务对于会员用户的吸引力是相同的，但此时完成相同任务的价格却明显降低。说明打包后不仅使任务的完成量得到提升，还可以节约 APP 平台成本。

5.3.2 多目标优化模型

接下来，通过多目标优化模型，对打包任务的价格方案作进一步的优化。

1、确定基础价格 G_b 和价格补偿值 ΔG_i

与仿真模型相同，将附件一中的地区划分为4部分。由于方案三中的价格未考虑地区的人口、经济、交通水平发展的平衡性，因此在方案三的基础 $M_{i2}=G_b$ 上增加价格补偿值 ΔG_i。最终价格 G_i 为基础价格与补偿值的和：$G_i=G_b+\Delta G_i(i=1,2,3,4)$。

2、确定区域的不平衡程度

每个地区在不同方面存在差异。一般认为，差异越高，不平衡程度越高，地方价格补偿值 ΔG_i 就越高。为减小不同地区之间的差异，因此在制定价格时考虑使不同区域的任务完成率 Q_i、任务难度 D_i、供求比率 P_i 数值较均衡。

以供求比率 P_i 为例，在某一固定时间，区域 i 中每个单独任务或打包任务的平均供求比率 P_i 为该区域所有任务 n_i 与会员人数 m_i 的比值。即

$$P_i=n_i/m_i \tag{27}$$

当区域 i 中任务的平均供求比率 P_i 越高时，P_i 与4个区域平均供求比 $\overline{P_i}$ 的比值 $P_i/\overline{P_i}$ 也越高，此时 APP 平台的定价便越高，二者为正相关，且比值一定。因此得到：

$$\begin{cases} \dfrac{\Delta G_1}{P_1/\overline{P_i}}=\dfrac{\Delta G_2}{P_2/\overline{P_i}}=\cdots=\dfrac{\Delta G_9}{P_9/\overline{P_i}} \\ n_1\Delta G_1+n_2\Delta G_2+\cdots+n_4\Delta G_4=n_{总}\overline{\Delta G_i} \end{cases} \tag{28}$$

以上为理想情况。结合实际，我们假设价格的补偿值G_i在 0.5 元—3.5 元之间。此时等式不一定成立，为保证各区域的供求比较均衡，将方差控制在最小即可。任务完成率Q_i、任务难度D_i同理。与此同时，对于 APP 方，希望把任务的定价控制在最小值。根据以上分析建立多目标的优化模型：

$$\min\begin{cases}\sum_{i=1}^{4}\Delta G_i \\ \sum_{i=1}^{4}\left(\frac{\Delta G_i}{P_i/\overline{P_i}}-\frac{\overline{\Delta G_i}}{\overline{P_i}/\overline{P_i}}\right) \\ \sum_{i=1}^{4}\left(\frac{\Delta G_i}{\overline{Q_i}/Q_i}-\frac{\overline{\Delta G_i}}{\overline{Q_i}/Q_i}\right) \\ \sum_{i=1}^{4}\left(\frac{\Delta G_i}{D_i/\overline{D_i}}-\frac{\overline{\Delta G_i}}{\overline{D_i}/\overline{D_i}}\right)\end{cases} \tag{29}$$

在该模型中，不仅结合实际将补偿价格控制在一定区间内，而且需区分单独任务和打包任务的价格，假设区域i中，打包任务的比例为c_{i1}，补偿金额为ΔG_{i1}单独任务的比例为c_2，补偿金额为ΔG_{i2}。因此约束条件为：

$$s.t.\begin{cases}0.5<G_i<3.5 \\ k_1G_{i1}=k_2G_{i2} \\ c_1+c_2=1 \\ c_1\Delta G_{i1}+c_2\Delta G_{i2}=\Delta G_i\end{cases} \tag{30}$$

3、定价方案确立

将附件一中的地区划分为4部分，如图 13 所示：

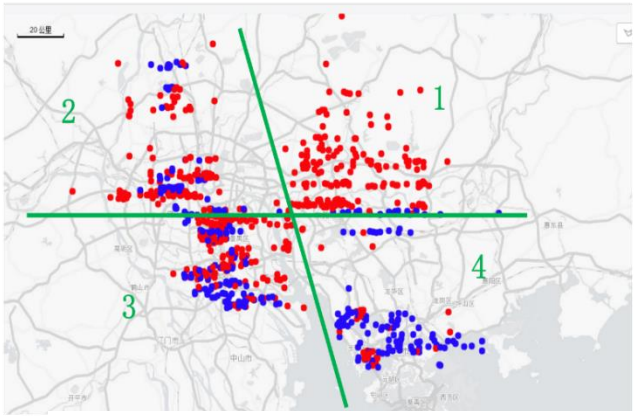

图 13 地区的划分情况

根据多目标优化模型求出 835 个新的价格方案，作为方案四。求解出这 4 个区域在将任务单独发布及打包发布的补偿值，如表 5 所示：

表 5 各区域打包任务和单独任务的补偿值

	区域 1	区域 2	区域 3	区域 4
打包任务	0.58	0.95	1.24	2.32
单独任务	1.24	0.88	1.33	1.85

并得到各个区域的任务完成情况。如图 14 所示：

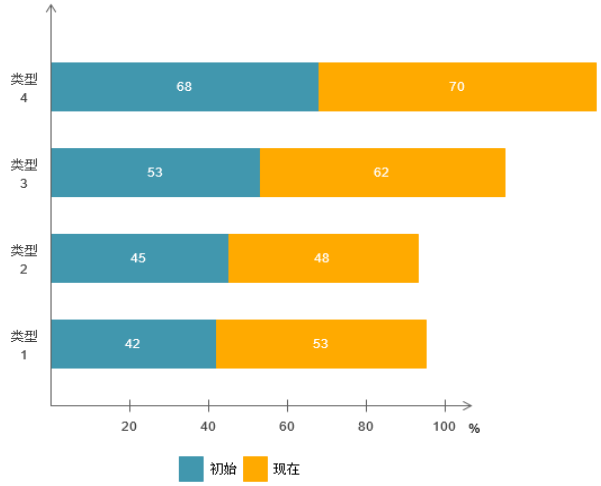

图 14 不同区域任务打包前后的完成率

由图 14 可知，4 个区域任务打包发布比单独发布的完成率更高，说明在这三个区域内，打包发布不仅避免了用户争相选择任务，并且可以进一步改善任务的完成情况，提高了完成任务的效率。

4 个区域任务打包发布与单独发布相比，提高了完成任务的效率，与多目标优化模型的结果一致。

4.灵敏度分析

记录随着补偿值的变化，各区域指标的变化率，如图 15 所示。

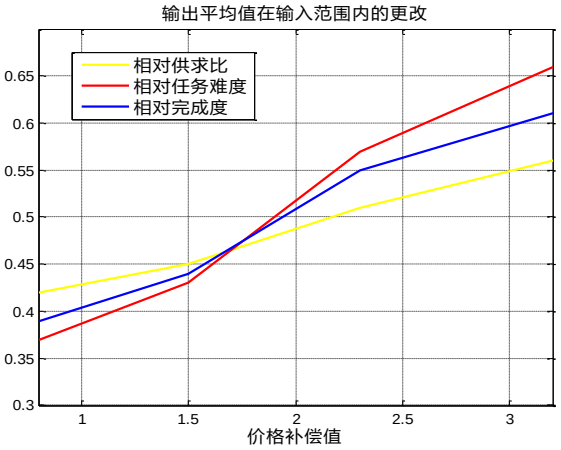

图 15 各个参数的灵敏度分析

由图 15 可以看出，对价格补偿值灵敏度由大到小分别为：相对任务难度，相对任务难度和相对供求比。也就是说，价格补偿值对缓解由任务难度造成的各区域间的不平衡效果最为明显。

5.4 问题四：基于 C-means 聚类的定价模型

5.4.1 打包方式的改进

在问题三中，我们选择通过网格对区域进行划分，将落入同一网格中的任务进行打包分配。这种方法具有较大的随机性，并且对落在网格边缘的任务划分造成困难。对该方法进行以下改进：

采取模糊 C-均值方法确定打包中心的位置 x_i。引入模糊数学中隶属度的概念。假定任务 j 对其打包中心位置 x_i 的隶属度为 v_{ij}，则

$$\sum_{i=1}^{2066} v_{ij} = 1, v_{ij} \in [0,1] \tag{31}$$

在聚类过程中，我们将对中心 x_i 隶属度较小的任务点进行调整，将其对聚类结果产生的影响降到最低。即将它们的隶属度调整为 $v_{ij}' = v_{ij} - \frac{(1-vij)}{2} v_{ij}$，对模型进行修正优化。最终经过查阅资料可知，打包中心点的位置为

$$x_{0i} = \frac{\sum_{i=1}^{2066} (v_{ij})^{\mathrm{m}} x_i}{\sum_{i=1}^{2066} (v_{ij})^{\mathrm{m}}} \tag{32}$$

其中，m 为常数，取 $m = 2$。在聚类过程中，我们的目的是使周围的点到聚类中心的距离达到最短值，因此建立目标函数：

$$\min | x_{0i} - v_{ij} | \tag{33}$$

采用 C-means 聚类算法[9]求解得到最佳聚类数 c 与各个打包中心点的位置。

5.4.2 价格的求解

经分析可知，附件三中的地点集中在广州市区、深圳沿海地区、深圳郊区三个位置。具体分布情况如图 16 所示：

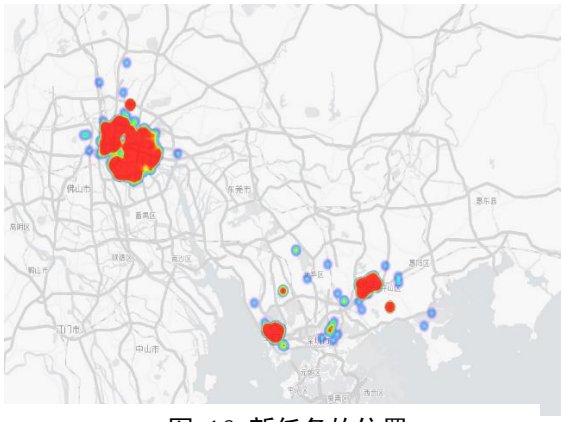

图 16 新任务的位置

在利用上述算法计算出打包中心点的位置后，在利用上述算法计算出打包中心点的位置后，考虑任务完成率 Q_i、任务难度 D_i、供求比率 P_i 先后利用前三问中的动态演化博弈模型、微观动态仿真模型和多目标优化模型，产生三种定价方案，分别记为方案一、方案二、方案三。具体的过程如下：

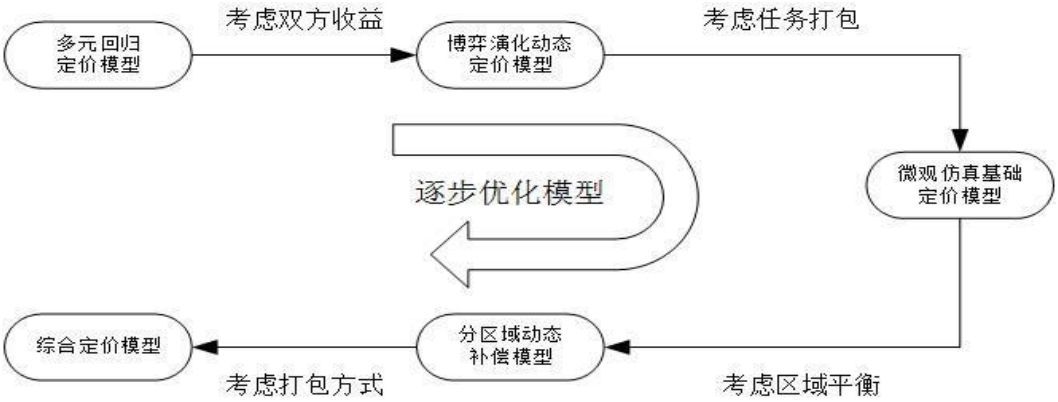

图 17 具体的价格制定优化方案

5.4.3 评价定价方案的实施效果

1、根据不同方案中任务的完成率进行对比

随着方案二到方案四，我们依次从会员与 APP 平台收益的博弈关系、打包任务与单独任务吸引能力、区域之间的发展平衡三个方面对定价方案进行优化调整。求解出任务的完成率 $Q_i(i=1,2,3)$，如图 18 所示:

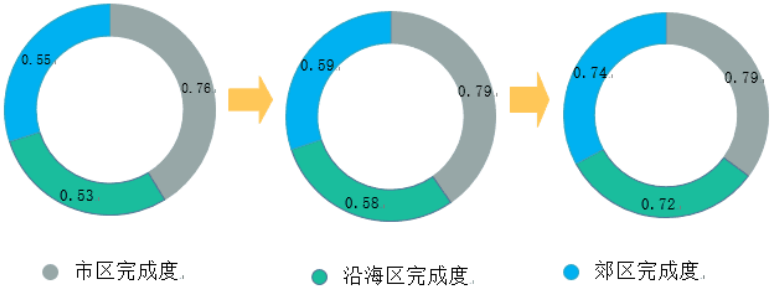

图 18 不同方案的完成率

可以看出，随着利用前三问中的动态演化博弈模型、微观动态仿真模型和多目标优化模型对定价方案进行优化的过程中，任务的完成率在不断增加。但是通过多目标优化模型建立的价格方案四中，不同地区的任务完成率更加均一些，这是由于只有多目标优化模型中考虑到了不同地区的不平衡度，此时的定价方案更合理。

2、根据不同地区对价格方案进行对比

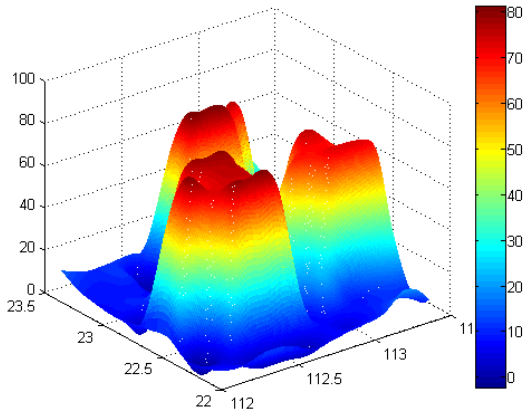

图 19 不同地区的价格对比

在对定价方案进行优化后，我们得到了如图 19 所示的定价方案。从图中可以清楚地看出定价被分成了三大部分，经分析可知，这三部分分别对应于广州市区、深圳沿海地区、深圳郊区三个位置。由图可知，广州市的平均价格是最高的，这是因为广州市经济状况良好，交通水平较高，并且任务的投放密度比较高，因此价格较高，而深圳郊区的价格最低，较符合实际。并且可分析出，在三个位置的中心点处价格相对较低，这是由于位置中心的用户密度较大，且交通便利，因而任务的接受率本身较高，价格较低。

六、模型的评价

6.1 模型的优点

1、分析实际数据时，运用位置经纬度坐标巧妙地解决了路程问题。

2、第二问通过分析用户方和 APP 方的利益博弈关系，建立了动态演化博弈的定价模型，并推广出了均衡演化博弈模型和博弈收益均衡模型。利用均衡直线与均衡点解决了用户方与 APP 方的合作博弈问题，得到了合理的定价方案

3、仿真过程中提取部分城市道路动态仿真模拟，具备普适性。合理的假设使得计算简化易行。

4、依次从会员与 APP 平台收益的博弈关系、打包任务与单独任务吸引能

力的比较、区域之间的发展平衡三个方面对定价方案进行优化调整，建立多个模型，由浅入深，层层递进，使定价方案更具说服力。

6.2 模型的缺点：

1、由于城市本身数据的不确定性，大部分指标以均值来统计，与实际问题的最终结果有一定差别

2、为了简化计算，相关因素的考虑做了适当修正，带有一定的主观性。

七、模型的推广

1、本文的演化博弈模型主要用于研究 APP 和会员之间的收益博弈，还可以进一步研究 APP 与公司双方关于 APP 提成的博弈，APP 之间存在的竞争博弈。建立一个多方综合博弈的模型。

2、本文建立了微观仿真模型用来确定打包后的任务价格，可以建立包含多个微观仿真的宏观模型，对所有位置的任务完成情况进行模拟。

八、参考文献

[1]根据两点经纬度计算距离，http://www.cnblogs.com/ycsfwhh/archive/2010/12/20/1911232.html2017 年 9 月 16 日。

[2]Spearman 秩相关检验，https://wenku.baidu.com/view/bbe42447cc7931b765ce15f6.html?qq-pf-to=pcqq.group，2017 年 9 月 17 日。

[3]许凤华李述山张英，《基于双重筛选的多因变量偏最小二乘逐步回归法》，《统计与决策》，2008 年。

[4]王凯，刘财，Pearson 相关系数法快慢横波波场分离[J]，2012 年 6 月，第 31 卷，第 2 期。

[5]周诗灿，梁帅，吴丽丹，葡萄酒质量的评价[C], 2015 年，12-21 页。

[6]张新华，赖明勇，基于演化博弈的电力竞价市场均衡分析[J]，2008 年，第 2 期。

[7]王一帆，基于打车软件的出租车服务模式优化研究[C]，2014 年。

[8]王嘉，模糊 C-均值算法在拼车系统中的应用[C]，2008 年 12 月 1 日。

[9]朱勇，基于 MATLAB 的 BP 神经网络应用，https://wenku.baidu.com/view/bf69f77d31b765ce05081479.html，2017 年 9 月 17 日。

附录

附录一

各城市的指标值：

	广州	深圳	佛山	东莞
总面积	19610.94	19492.6	8630	6827.67
人均 GDP(万美元)	14.5	17.1	11.6	8.2
人口(万)	1350.11	1137.89	743.06	825.41
交通水平		0.8		

附录二

回归模型的结果

	非标准化系数		标准系数	t	显著性	B 的 95.0%置信区间	
	B	标准错误	贝塔			下限值	上限值
常量	71.736	1.699		42.232	0	68.364	75.107
任务密度系数	-0.399	0.123	-0.382	-3.234	0.002	-0.644	-0.154
与市中心距离系数	0.035	0.045	0.09	0.766	0.446	-0.055	0.125

附录三：

Matlab 主程序：基于 MATLAB2014a 编程 相关数据详见附录

程序一：处理相关数据 经纬度矩阵转距离矩阵

```
function cumcmt1
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here data process
%子函数    [geoid,msg] = geoidtst(geoid)
%         shortdistance(lat1, lon1, lat2, lon2, ellipsoid)
tic
disp(sprintf('正在载入相关数据,请耐心等待...'));
%%数据准备
distribute1=xlsread('C:\Users\song\Desktop\CUMCM2017\d1.xls');
distribute2=xlsread('C:\Users\song\Desktop\CUMCM2017\d2.xlsx')
;
number=length(distribute1);pi=3.1415926;
disp(['number is ',num2str(number)]);
x=distribute1(:,1);lati=x*(pi/180);
y=distribute1(:,2);loni=y*(pi/180);
distance=zeros(number,number);
ellipsoid=geoidtst(almanac('earth','ellipsoid'));
for i=1:number
```

```
        distance(i,i+1:number) =
shortdistance(lati(i)*ones(length(i+1:number),1),
loni(i)*ones(length(i+1:number),1),lati(i+1:number),
loni(i+1:number),ellipsoid);
        distance(i,1)=distance(i,1)*6371*1000*2*pi/360;
    end
    dis=(distance+distance')/10;
    distribute1=distribute1(:,1:2);
    distribute2=distribute2(:,1:2);
    diss=[distribute2;distribute1];
    number2=1877+835;
    disp(['number2 is ',num2str(number2)]);
    xx=diss(:,1);latii=xx*(pi/180);
    yy=diss(:,2);longii=yy*(pi/180);
    distance2=zeros(number2,number2);
    ellipsoid = geoidtst(almanac('earth','ellipsoid'));
    for i=1:number2
        distance2(i,i+1:number2) =
shortdistance(latii(i)*ones(length(i+1:number2),1),
longii(i)*ones(length(i+1:number2),1),latii(i+1:number2),
longii(i+1:number2),ellipsoid);
        distance2(i,1)=distance2(i,1)*6371*1000*2*pi/360;
    end
    disfarm=(distance2+distance2')/10;
    dis2=disfarm(836:2712,1:835);
    xlswrite('C:\Users\song\Desktop\CUMCM2017\passenger.xlsx',dis)
;
    xlswrite('C:\Users\song\Desktop\CUMCM2017\guanxi.xlsx',dis2);
    toc
    程序二:子函数1：function [geoid,msg]=geoidtst(geoid)
    程序三:子函数2：function rng=shortdistance(lat1, lon1, lat2, lon2,
ellipsoid)
    程序四：求任务密度
    function [ret]=passenger_density()
    distribute=xlsread('C:\Users\song\Desktop\CUMCM2017\passenger.xls
x');
    sum=0;
    data=distribute;
    cnt=zeros(835,1);
    for i=1:835
        sum=0;
        for j=1:835
            if data(i,j)<1.5299
                sum=sum+1;
```

```
            end
        end
        cnt(i,1)=sum;
    end
    xlswrite('C:\Users\song\Desktop\CUMCM2017\ÈÎÎñÃÜ¶È.xlsx',round(cn
t/10));
    程序五：求会员密度
    function [ret]=huiyuan_density()
    distribute=xlsread('C:\Users\song\Desktop\CUMCM2017\guanxi.xlsx')
;
    data=distribute';
    cnt=zeros(835,1);
    for i=1:835
        sum=0;
        for j=1:1877
            % %d=sqrt(s^2*exp(-lambda)*k/a*pi);一般k=0.8,s为城市面
积,lambda人均GDP,a总人口-->d=1.5299
            if data(i,j)<1.5299
                sum=sum+1;
            end
        end
        cnt(i,1)=sum;
    end
    xlswrite('C:\Users\song\Desktop\CUMCM2017\¹ØÏµÃÜ¶È.xlsx',round(cn
t/10));
    程序六：求离市中心的距离
    function [ret]=shizhongxin_dis()
    distribute=xlsread('C:\Users\song\Desktop\CUMCM2017\zhongxin.xlsx
');
    x=zeros(835,1);
    for i=1:835
        data=distribute(i,:)
        x(i,1)=min(data);
    end
    xlswrite('C:\Users\song\Desktop\CUMCM2017\ÊÐÖÐÐÄ¾àÀë.xlsx',x);
    程序七：聚类分析 julei.m
    程序八：动态博弈论演化
    function  drawItems(AB,Q,init,li,lj,k)
    Aa = AB(1,1);
    Ab = AB(1,2);
    Ba = AB(2,1);
    Bb = AB(2,2);
    for i=0:0.1:li
        for j=0:0.1:lj
```

```
            [T,Y]=ode45('fx',init,[i j],[],Q,Aa,Ab,Ba,Bb);
            grid on
            figure(k)
            hold on
            plot(Y(:,1),Y(:,2));
        end
    end
```

程序九：BP神经网络预测

```
    clc;clear all;
    disp(sprintf('正在载入相关数据,请耐心等待...'));
    x=xlsread(' C:\Users\song\Desktop\CUMCM2017\附件一.xls','Sheet1');
    x=x(1:835,1:3);p=x';% %输入矩阵
    y=xlsread(' C:\Users\song\Desktop\CUMCM2017\附件一.xls','Sheet1');
    y=y(1:835,4);t=y';
    [pn,minp,maxp,tn,mint,maxt]=premnmx(p,t);
    dx=[-1,1;-1,1;-1,1]; %归一化处理后最小值为-1，最大值为 1
    %%BP 网络训练
    net=newff(dx,[3,5,1],{'tansig','tansig','purelin'},'traingdx'); %
建 立 模型，并用梯度下降法训练
    net.trainParam.show=10000; % 10000 轮回显示一次结果
    net.trainParam.Lr=0.05; %0.05 学习速度
    net.trainParam.epochs=20000; %最大训练轮回为 20000 次
net.trainParam.goal=0.065*10^(-2); %均方误差
    net=train(net,pn,tn); %开始训练，其中 pn,tn 分别为输入输出
    an=sim(net,pn); %用训练好的模型进行仿真
    a=postmnmx(an,mint,maxt); %把仿真得到的数据还原为原始的数量级;
    pnew=xlsread(' C:\Users\song\Desktop\CUMCM2017\附件
一.xls','Sheet2');
    pnew=pnew(1:835,1:3);pnew=pnew'
    pnewn=tramnmx(pnew,minp,maxp);%利用原始输入数据的归一化参数对数据归一
化；anewn=sim(net,pnewn);
    anew=postmnmx(anewn,mint,maxt) %把仿真得到的数据还原为原始的数量级;
    anew=round(anew)';
```

程序十：微观仿真代码

```
    clear all;clc;
    tic;
    disp(sprintf('正在载入相关数据,请耐心等待...'));
    % %随机生成80个订单 passenger   随机生成200个会员  随机生成5个中心点center
    huiyuan_data=[];data=[];data1=[];
    passenger=80;huiyuan=200;center=5;
    p=12;
    passenger_lati=22+p*rand(passenger,1);
```

```
passenger_loni=112+p*rand(passenger,1);
huiyuan_lati=22+p*rand(huiyuan,1);
huiyuan_loni=112+p*rand(huiyuan,1);
center_lati=22+p*rand(center,1);
center_loni=112+p*rand(center,1);
plot(passenger_lati,passenger_loni,'bo','MarkerSize', 5);
axis([20 35 111 124]);
hold on
plot(huiyuan_lati,huiyuan_loni,'g*','MarkerSize', 5);
axis([20 35 111 124]);
hold on
plot(center_lati,center_loni,'mp','MarkerSize', 8);
axis([20 35 110 125]);
xlabel('纬度')
ylabel('经度')
legend('任务','会员','中心点');
for ii=110:125
    x=20:1:35;y= 0*x+i;
    if mod(ii,5) == 0
       plot(x,y,'k:');
    end
end
for jj=21:35
    x=110:125;y=0*x+jj;
    if mod(jj,5) == 0
       plot(y,x,'k:');
    end
end
passenger_data=[passenger_lati passenger_loni ones(passenger,1)];
huiyuan_data=[huiyuan_lati huiyuan_loni ones(huiyuan,1)];
center_data=[center_lati center_loni ones(center,1)];
data=[passenger_data;center_data;huiyuan_data];
dis=[];dis=data;
nfarms=passenger+center;
disp(['nfarms is ',num2str(nfarms)]);
k=size(dis,1);
dis_x=repmat(dis(:,1),1,k);
dis_y=repmat(dis(:,2),1,k);
dist_x=repmat(dis(:,1)',k,1);
dist_y=repmat(dis(:,2)',k,1);
distance=sqrt((dis_x - dist_x).^2+(dis_y - dist_y).^2);
%提取距离矩阵
passenger_dis=distance(1:passenger,1:passenger);
center_dis=distance(passenger+1:passenger+center,passenger+1:pass
```

```
enger+center);
    %中心点到任务点的距离
    dis_cp=distance(passenger+1:passenger+center,1:passenger);
    %提取人到每个任务的距离矩阵
    dis_hr=distance(passenger+center+1:end,1:passenger+center);
    %为了简化 使每个包中点个数相同
    number=10;[m,n]=size(dis_cp);
    [Y,I]=sort(dis_cp,2,'ascend');   %按照行进行升序排列
    RowCol.value=Y(:,1:number);      %结构体RowCol，存放数值
    %RowCol.row=repmat((1:N).',number);  %行坐标值
    RowCol.col=I(1,1:number);        %列坐标值
    for m=1:center
        aa{m}=I(m,:);
    end
    for mm=2:center
        for n=1:number
            for nn=mm:center
            aa{nn}(find(aa{nn}==aa{mm-1}(n)))=[];
            end
        end
    end
    center_good=[aa{1}(1:number); aa{2}(1:number); aa{3}(1:number);
aa{4}(1:number); aa{5}(1:number)];%5*10
    %假设打包项目和未打包项目对人的吸引力相同只要落入会员范围之内就会去选择 然后开
始做
    speed_do=6000*rand();
    r_valid=500*rand();%距离意愿 单位为m
    speed=0;
    %人如果做得是打包的任务 那么要将包内任务做完才能去接下一个任务
    all_xuhao=[1:passenger+center]';
    all_zuobiao=[passenger_data;center_data];
    passenger_xuhao=all_xuhao(1:passenger);
    center_xuhao_1=[101 102 103 104 105];
    center_xuhao_2=[aa{1}(1:number);aa{2}(1:number);aa{3}(1:number);a
a{4}(1:number);aa{5}(1:number)]';
    center_xuhao=[center_xuhao_1;center_xuhao_2];
    %´ò°ü³ÉÎåÀà
    for i=1:length(center_xuhao)
        passenger_xuhao(find(passenger_xuhao==center_xuhao(i)))=[];
    end
    bao_nei=center_xuhao;
    bao_wai=passenger_xuhao;
    c=0;
    data1=[passenger_data;center_data];
```

```
    huiyuan_data(:,3)==0;data1(:,3)==0;renwu_data=data1;
    all_K = [];xmax = 111*cos(pi*34/180)*1.4;
    ymax = 0.7*111;
    for i=1:24*3600
    r_valid=500*rand();
    lc =huiyuan_data(:,3)-0.1;
        lc(lc<=0)=0;
        huiyuan_data(:,3)=lc;
        valid_lines=find(lc==0);
        for m=1:length(valid_lines)
            k = valid_lines(m);
            for e=1:100
                degree = 2*pi*rand();
                new_x = huiyuan_data(k,1) + 0.1.*cos(degree);
                new_y = huiyuan_data(k,2) + 0.1.*sin(degree);
                if(new_x>=0&& new_x<= r_valid&& new_y>=0 &&
new_y<=r_valid)
                huiyuan_data(k,1:2) = [new_x,new_y];
                   break
                end
            end
        end
         for j=1:passenger+center
             ren_num=find(dis_hr(m,j)<r_valid);
             if isempty(ren_num)           continue;
             else
                c=j;
                if ismember(c,bao_nei(:,1));
                   for ii=1:11
                      renwu_data(bao_nei(ii,1),3)=0;
                   end
                   speed=speed_do*11+25;
                elseif ismember(c,bao_nei(:,2));
                   for ii=1:11
                      renwu_data(bao_nei(ii,2),3)=0;
                   end
                   speed=speed_do*11+25;
                elseif ismember(c,bao_nei(:,3));
                   for ii=1:11
                      renwu_data(bao_nei(ii,3),3)=0;
                   end
                   speed=speed_do*11+25;
                elseif ismember(c,bao_nei(:,4));
                   for ii=1:11
```

```
                    renwu_data(bao_nei(ii,4),3)=0;
                end
                speed=speed_do*11+25;
            elseif ismember(c,bao_nei(:,5));
                for ii=1:11
                    renwu_data(bao_nei(ii,5),3)=0;
                end
                speed=speed_do*11+25;
            else
                renwu_data(j,3)=0;
            end
        end
    end
end
toc
```

程序十一：求解多目标优化 子函数：`function y=Fun(q)`

程序十二：求解多目标优化 主函数main.m

```
clear all;clc
fitnessfcn=@Fun;
 nvars=4;
 lb=[0.5,0.5,0.5,0.5];
 ub=[3.5,3.5,3.5,3.5];
 A=[];b=[];
 Aeq=[];beq=[];
options=gaoptimset('paretoFraction',0.3,'populationsize',100,'gen
erations',300,'stallGenLimit',200,'TolFun',...
    1e-10,'PlotFcns',@gaplotpareto);
[x,fval]=gamultiobj(fitnessfcn,nvars,A,b,Aeq,beq,lb,ub,options)
```

程序十三：不同地区的最终定价方案三维图 sanweitu1.m　　sanweitu2.m

程序十四:fangzhen3.m