校园帮外卖代取系统课程报告

杨如帅，1190200803

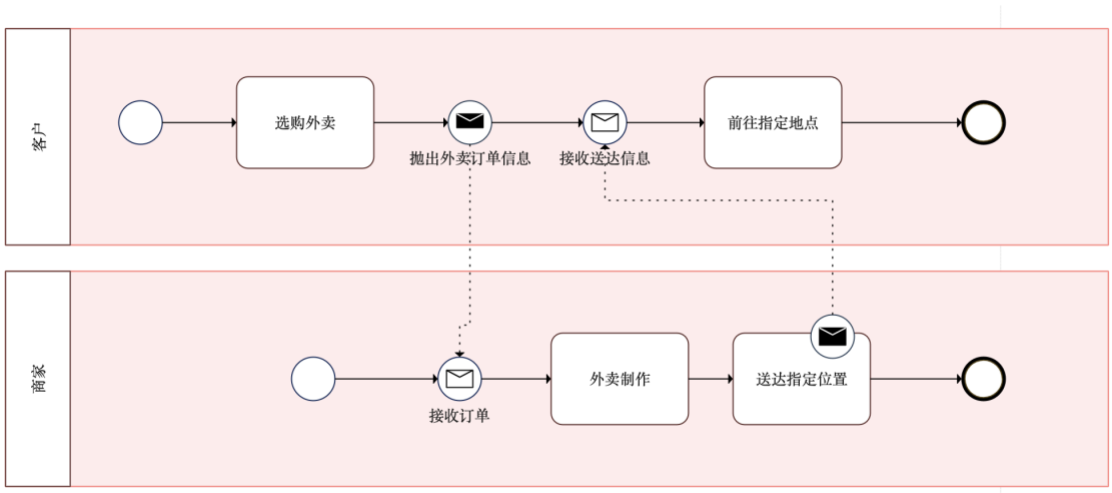
摘 要：该报告旨在针对大学校园内外卖配送效率低下的问题，运用业务流程管理的方法对系统进行业务流程创新，提高了外卖服务"最后一公里"的配送效率。

关键词：校园外卖代取，服务创新，BPMN，微服务，BPEL。

1. 系统背景介绍

校园帮校园外卖代取系统旨在解决疫情期间学生的外卖取餐困难的问题。由于疫情，学生难以外出，学校食堂众口难调，只有通过选购外卖改善伙食。加之校园封闭，外卖小哥无法像过去那样一步到位送到宿舍楼下，学生需要步行至较远的校门口领取外卖。如遇天气状况较差的情况，取外卖路远且麻烦，效率很低。因此，解决校园外卖的“最后一公里”的业务流程问题迫在眉睫，需要提供一种配送代取的服务能够取代传统费时费力的业务过程。

系统背景介绍



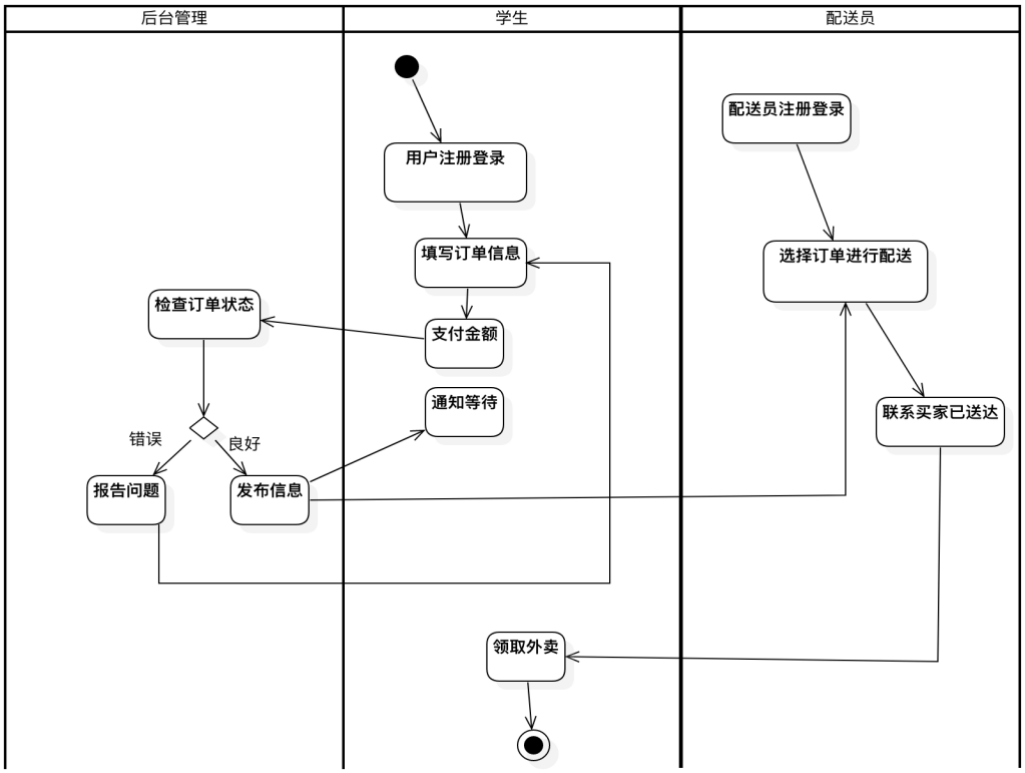
传统业务流程 as-is

2. 服务模式创新

2.1.基于互联网+外卖代取的服务创新：

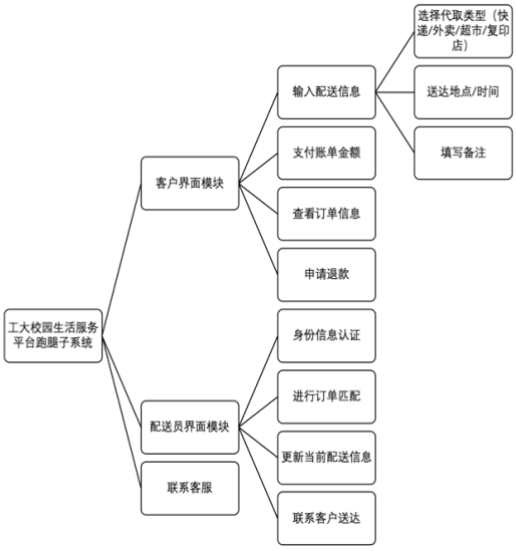
通过分析我们发现，之所以外卖服务存在效率低下的原因，本质上是因为外卖无法直接送到客户手中导致的，为了解决这个问题，我们需要考虑一种能够帮助客户解决校园外卖的“最后一公里”的业务流程。因此，下面提出一种新的配送模式来更好的让客户足不出户也能快速高效地领取外卖。

我们可以考虑利用微信小程序，搭建一个校园服务外卖代取平台。学生可以在微信小程序上抛出自己的订单信息，校园内的配送员在小程序上接单并将外卖送达到学生手中。小程序的基本流程为：学生使用小程序需要进行用户注册，后方可进行购买、评论、收藏、与客服沟通、点击个人账号模块、个人消息、退出账号/登录新账号、点击登录，设置头像，昵称，绑定手机号等操作。普通用户通过登录微信小程序并提供校园外卖订单信息，外卖骑手经过身份认证后方可查看各订单情况并选择接单。骑手需要在限定的时间内根据用户提供的地址，外卖编号等信息在规定的时间内送达至指定地点。用户可以在订单途中和骑手联系或者修改外卖订单信息。上述流程概括起来如下图所示：



业务流程图

程序端设计原型图



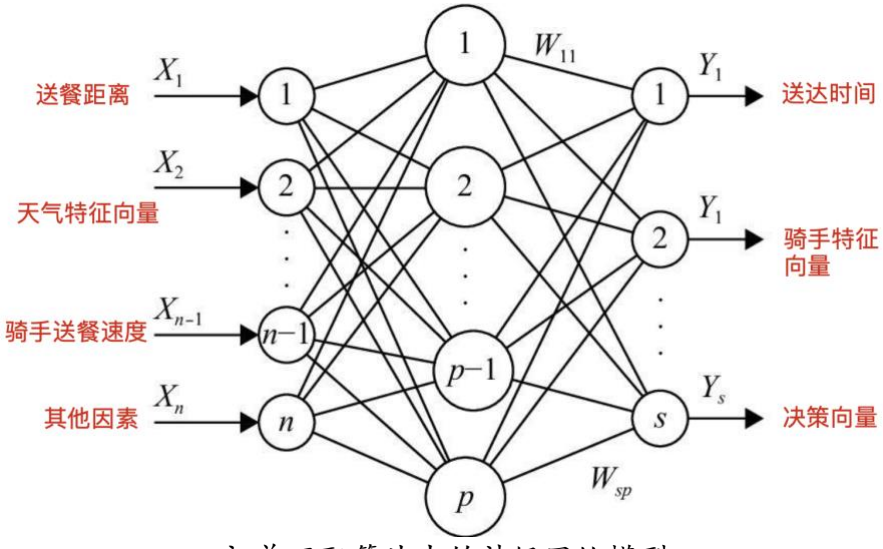
业务模块

2.2.基于对外卖订单配送时间优化的服务创新：

针对外卖订单配送服务而言，订单和服务提供方的匹配问题是一个非常关键的问题。我们最初的想法是采用依赖骑手抢单模式或者人

工派单模式。抢单模式的优势是开发难度低，服务提供者（骑手）的自由度较高，可以按照自身的需要进行抢单，但其缺点也很明显：骑手/司机只考虑自身的场景需求，做出一个局部近优的选择，然而由于每个骑手掌握的信息有限又只从自身利益出发来决策，导致配送整体效率低下，从用户端来看，还存在大量订单无人抢或者抢了之后造成服务质量无法保证（因为部分骑手无法准确预判自己的配送服务能力）的场景，用户体验比较差。而人工派单的方式，从订单分配的结果上来看，一般优于抢单模式。在订单量、骑手数相对比较少的情形下，有经验的调度员可以根据订单的属性特点、骑手的能力、骑手已接单情况、环境因素等，在骑手中逐个比对，根据若干经验规则挑选一个比较合适的骑手来配送。一般而言，人工调度一个订单往往至少需要半分钟左右的时间才能完成，但是这样来说人工成本又太高，不可持续。

因此我们考虑使用机器学习模块负责从数据中寻求规律和知识，例如对商家到达时间、天气情况，平均骑行速度、骑行导航路径等因素利用神经网络来进行准确预估未来到达的时间、符合该订单配送要求的骑手（配送员）特征（不同的配送员具有不同的特征，例如所在位置，是否在配送途中，按时送达率等），调度决策相关信息。然后使用相似性匹配算法在配送员数据库中选取出满足这个订单需求的配送员进行配送。神经网络结构图如下：



订单匹配算法中的神经网络模型

网络输入是订单的相关信息和为了满足订单需求的一些必要条件，因此我们首先要构建对输入数据进行预处理。举例来讲，对于构建天气特征向量而言，我们考虑影响一些影响送达时间的因素，例如天气情

况（打雷，下雨，阴天），风向风力(微风，强风)，空气质量指数（良好，差）等，然后进行 One-hot 编码。然后考虑利用 20 * 6 * 6 * 8 的神经网络，分别表示一层输入层，两层中间层和一层输出层。激活函数我们采用 Rectified Linear Unit 函数，表达式为：

$$\text{relu}(x)=\max(x,0)$$

选择 ReLU 函数的原因 ReLU 的收敛速度要远远快于 sigmoid 和 tanh，在一定程序上缓解梯度消失的问题。算法的优化器我们选用 Adam（Adaptive Moment Estimation）优化器，更新规则如下：

$$v \leftarrow \beta_1 v + (1-\beta_1) g$$

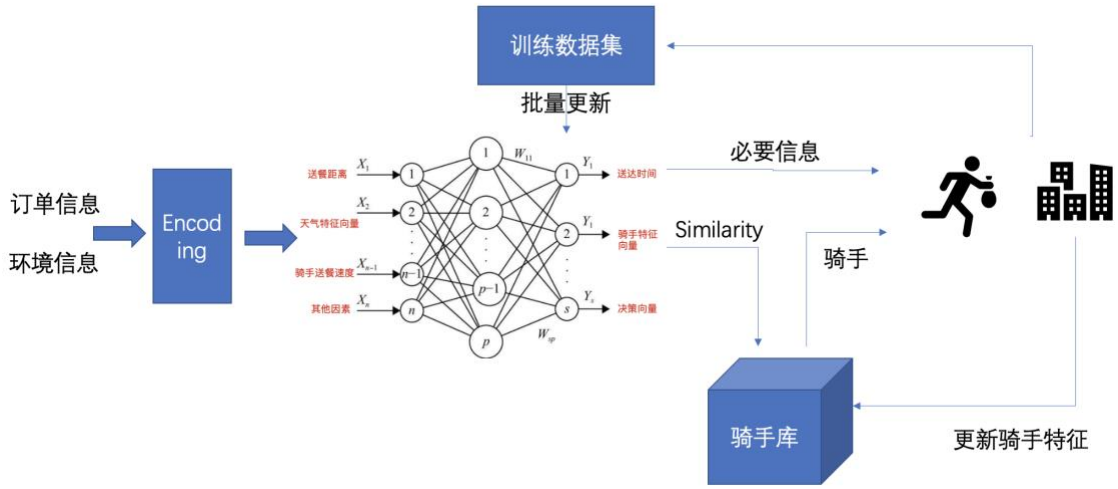
$$h \leftarrow \beta_2 h + (1-\beta_2) g \odot g$$

$$\theta \leftarrow \theta - \frac{\alpha}{\sqrt{h}} \cdot v$$

其中 v 是动量项，β是关于当前速度的一个衰减因子。给定一个数据集 $\{x, y\}_i$我们计算每一个 batch 的平均平方误差（MSE），然后利用上述更新算法来对网络参数进行更新，进而优化我们的订单匹配网络。最后，为了准确选择出具体的骑手，我们利用网络输出的骑手特征向量与骑手数据库中的数据计算余弦相似度，计算公式如下：

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}.$$

其中，A,B 分别为该订单需要的骑手特征和数据库中骑手的特征，余弦相似度越大，说明骑手越能匹配该订单的需求，因此我们从数据库中选择出相似度最高的骑手进行送餐即可。整个算法实现流程处理完毕。



模型流程图

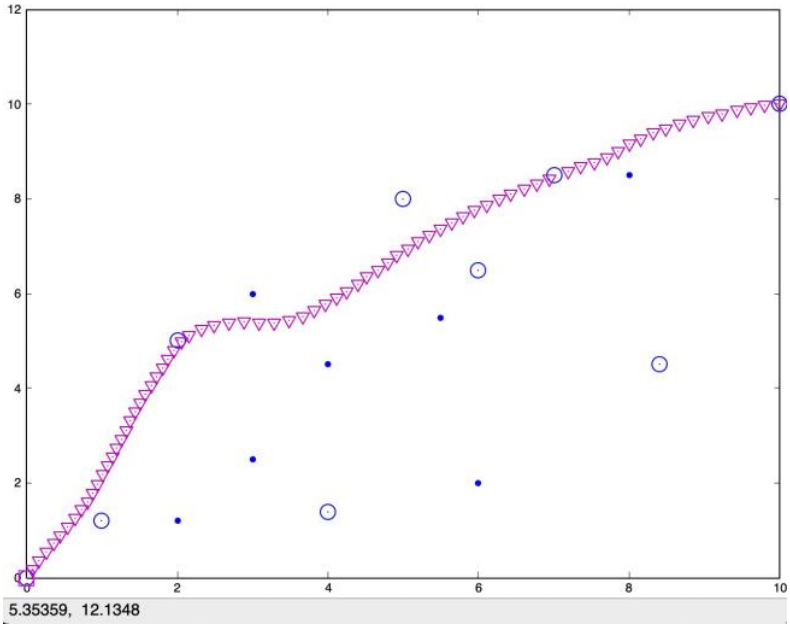
2.3.基于配送员路径规划的服务创新：

基于上外卖配送的服务中，送达时间效率是严重影响客户的满意度的主要因素。因此我们还需要考虑设计出一种能够计算出最短距离的路线为配送员和客户节省相应的时间。服务创新的设计思路如下：首先需要定义一组地图检查点集合，对于集合中任意一个元素，从地图上任意某一点出发，到达某一个目标地点的必要条件至少经过集合中的一个点。也就是说，我们给校园地图上某些路段做标记，使得这些点是整个路径的组成部分，我们引导配送员行驶在目标点的最短路径上。例如，我们调用的百度地图 API 来构建二维网状地图提供的导航，然后在配送员送达途中必须经过的路口设置检查点：

导航服务效果展示图，核心代码见附录

得到这个集合之后，我们为全局路径规划选取用改进的基于优先队列 A*算法生成一系列的目标点。然后利用人工势能的方法，对判断起始点和中间点连成的直线上是否有障碍物/建筑物，若没有，则直接生成直线路径，否则，避开障碍物集中区使得在靠近建筑物时能留有

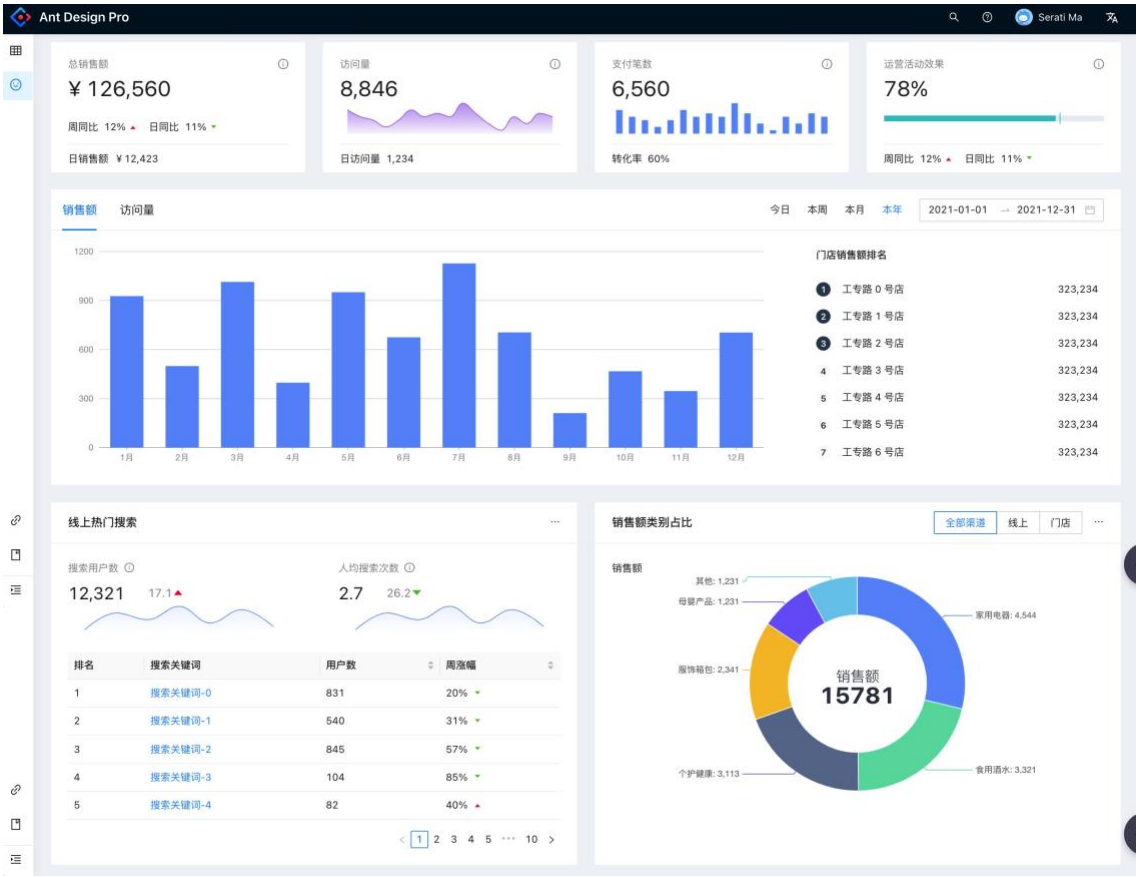
足够的空间。我们通过二维仿真建模得到算法效果如下：



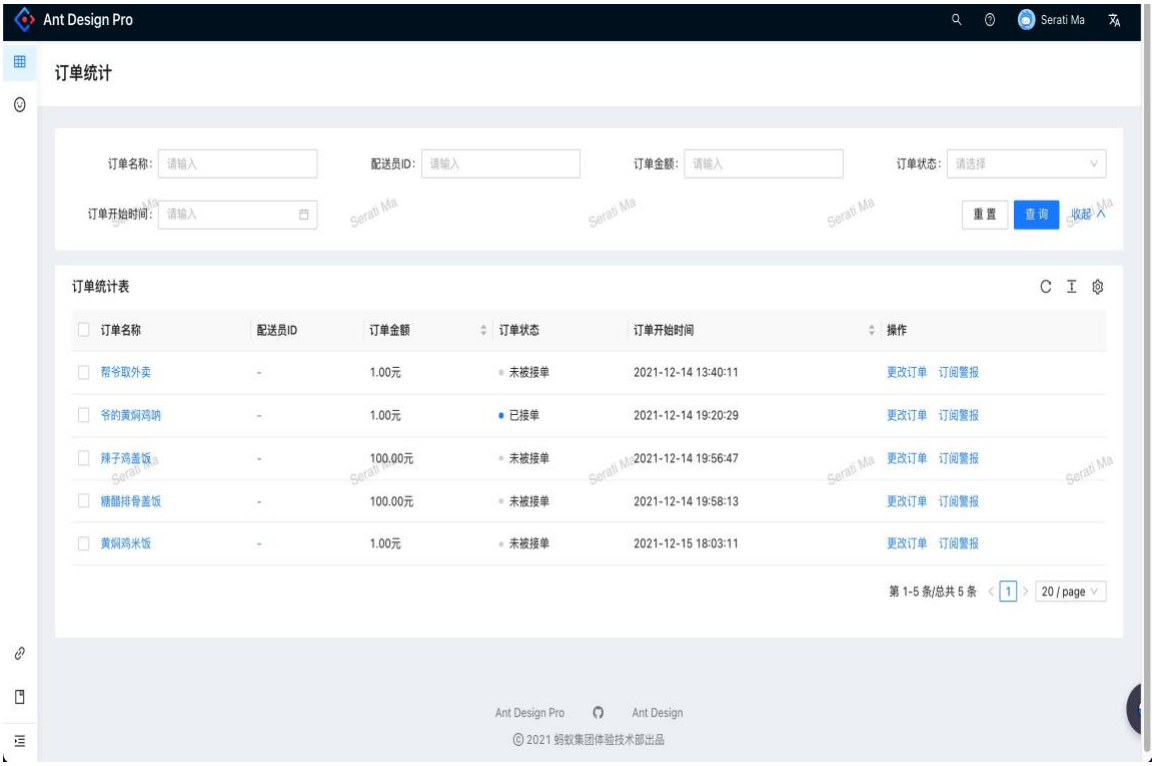
最短路径规划算法示意图。为了清晰地描述该算法，这里省略了地图背景，其中实心点是建筑物，最左和最右的圆点分别是外卖所在地和客户所在地，其他圆点为地图上标记的关键点。红线形成的路径即为地图上的最短路径，代码见附录

2.4.基于资金冲突避免的业务流程创新

为了解决业务冲突服务的问题，现有的业务结构（见 As-is 图）是没有第三方后台管理介入的，这样的结果会存在一些问题。例如，当学生和配送员发生冲突或者一些涉及人身风险的事件（恶意订单、利用订单挂广告等），或者出现外卖遭受损坏或者丢失，这时候可能需要后台管理员的介入来协调解决矛盾，我们设计出相应的 Web 服务平台提供给系统管理员来监听和管理交易过程中可能出现的资金流和订单冲突问题：



管理员统计订单数据

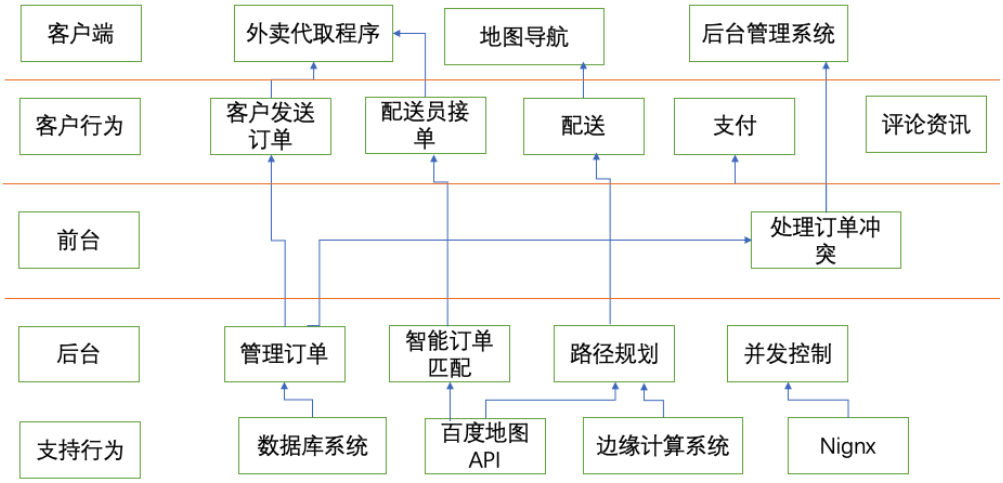


管理员订单管理

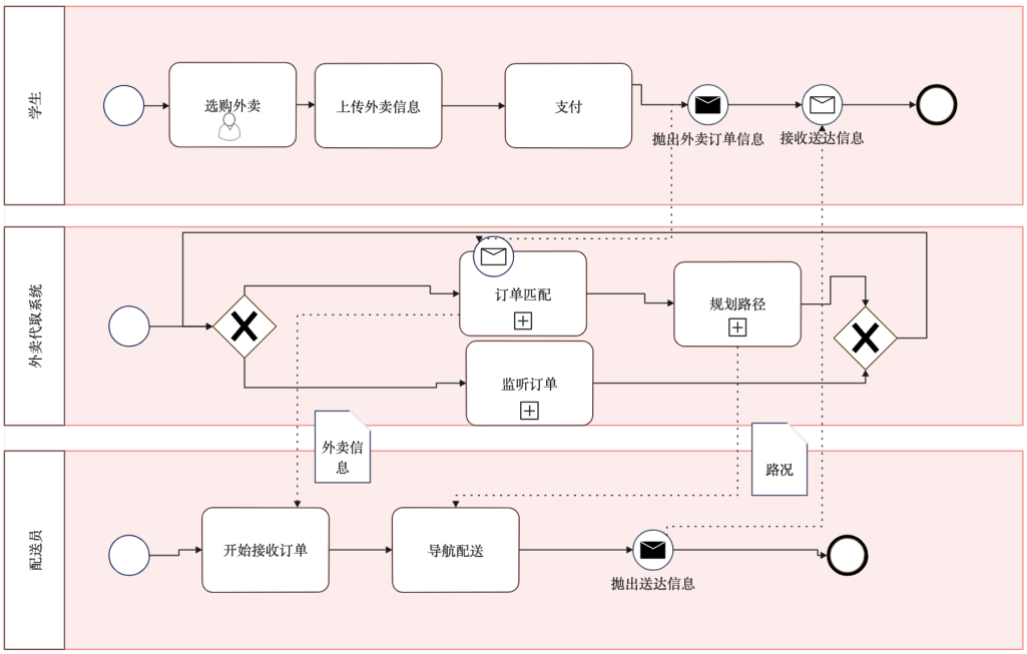
3. 系统建模分析

3.1.业务流程建模（四个核心业务流程建模）

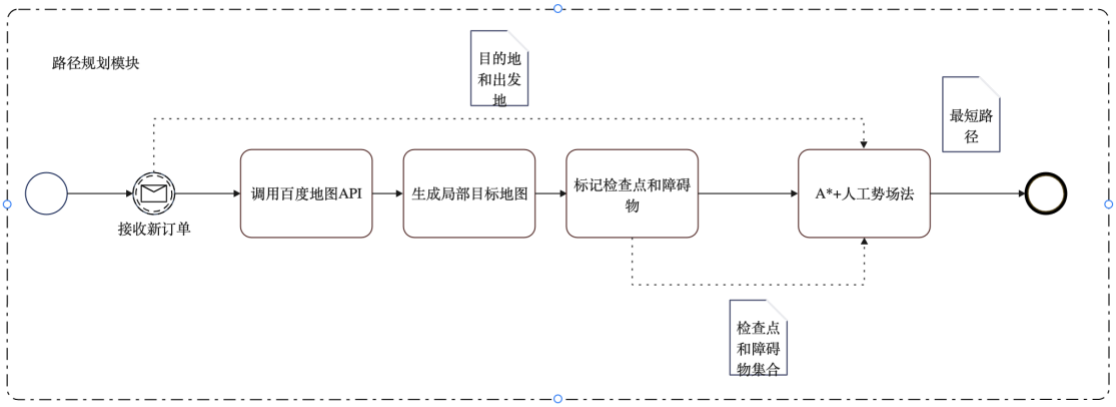
上述通过对现有的业务分析得到了一系列的服务创新模式，根据这些新的服务我们设计出服务蓝图如下：



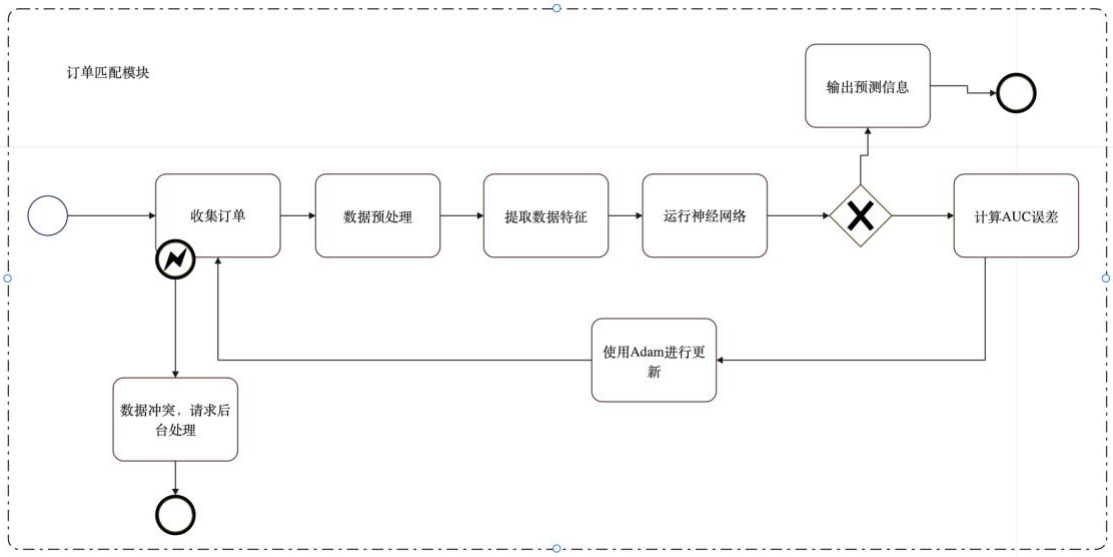
可以发现通过自动化传统的路径规划和订单匹配模块，从服务创新上极大地解放了前台的业务人员的工作量，只需要少量的管理员工即可完成对该外卖配送系统的管理。我们 BPMN 建模如下图所示：



现有业务流程 BPMN to-be 图



细化路径规划模块 BPMN

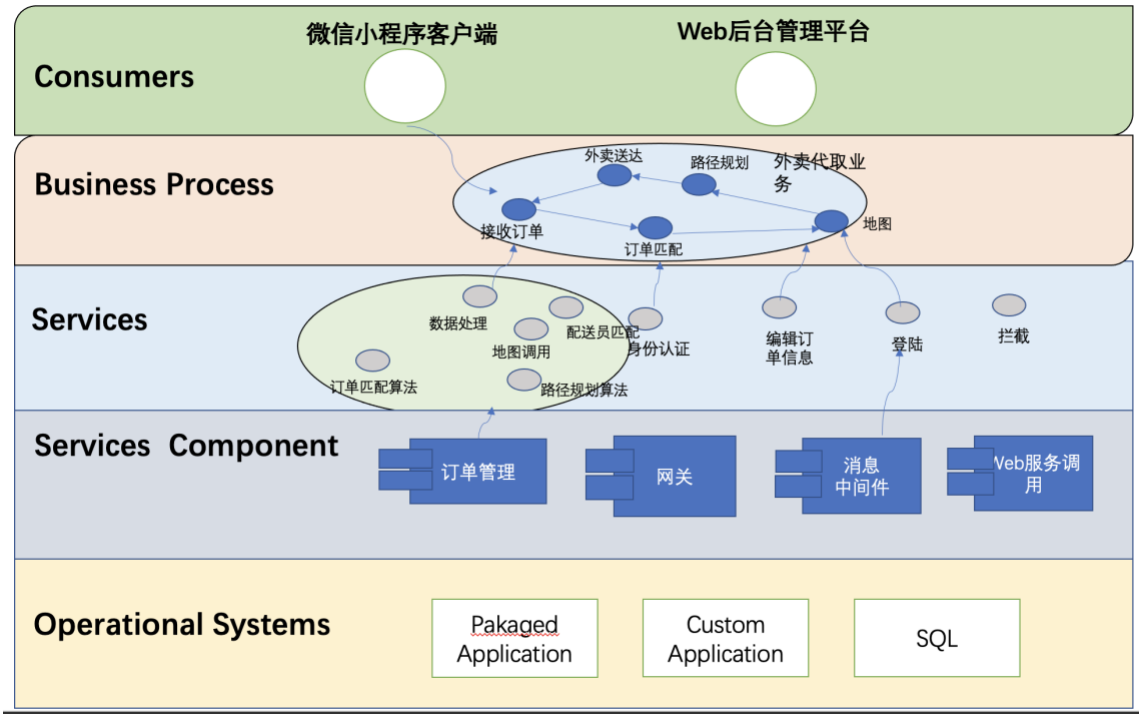


细化订单匹配模块 BPMN

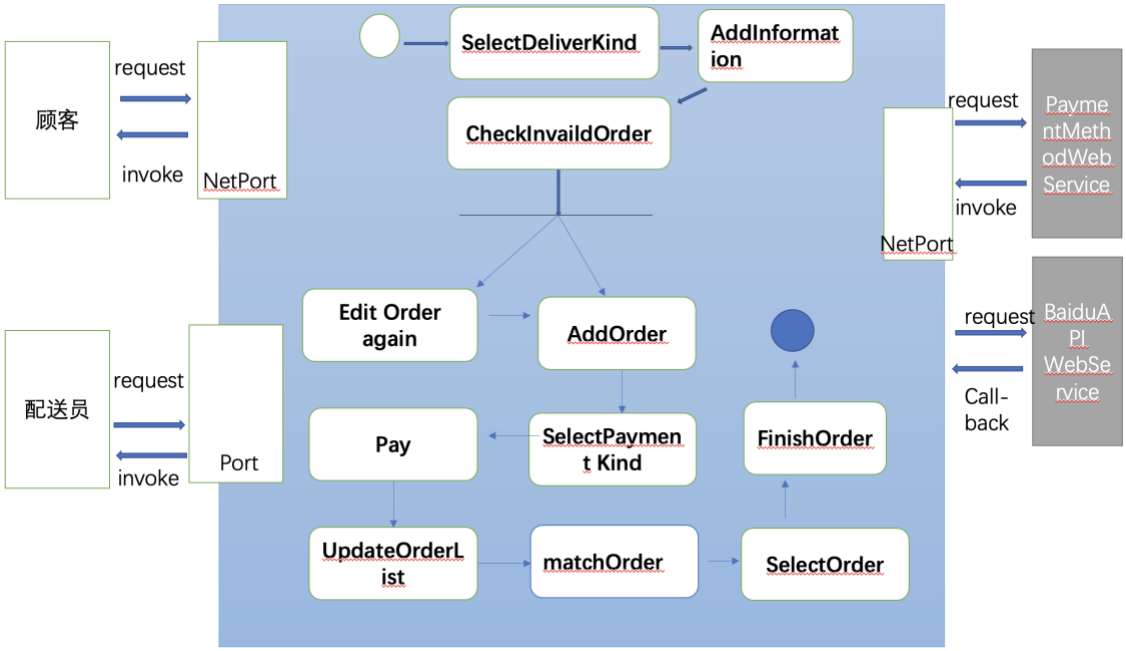
3.2.服务架构设计和服务组合：

跟据对业务流程进行分析之后，我们开始设计相应的系统架构，前端采用的 *AntDesignPro* 和微信小程序 Wechat Dev，后端采用 Spring boot，mybaits 等一系列 Java 后端开发技术，我们考虑利用 SOA 的设计理念来对整个服务系统进行设计，我们将业务流程拆分成一系列能够提供一系列完整功能的服务集群，为了使得服务能尽量解耦，我们采用 Dubbo+ zookeeper 的框架来搭建系统后端，将单一服务放至不同的 docker 容器平台并利用 k8s 来进行管理；再服务注册到 zookeeper 服务注册中心然后客户端从注册中心调用相应的服务来组合形成相应的业务流程。相应的服务设计拆分和设计已放至**附录**中，以我们系统中的主要业务流程外卖代取系统为例，描述上述系统架构设计和服务组合

如下：



系统分层架构设计图



服务组合模型

4. 总结：

通过对以往低效的外卖代取服务进行了业务流程分析，发现了流程中存在的配送效率低下的问题并进行了业务创新，通过构建新型的外卖代取服务平台以及对其中订单匹配，路径规划和资金流的冲突管理进行了业务创新的改进。具体来讲，通过输入订单和当天环境的相关特征信息至训练好的神经网络中，得到该订单所需配送员的相关特征，然后从配送员资料库中对特征进行匹配，选出最适合的配送员进行配送；再调用百度地图 APi，通过 A* + 人工势能的方法进行路径规划选择出最短路径供配送员进行配送，以期减少不必要的时间开销。以上四种主要的创新优化方法使得整个业务系统的效率得到了改进，也减轻了诸多的人力成本，具有很好的借鉴意义。

5. 附录：

表 1

系统名称	业务流程名称	业务子流程名称	服务名称	初始服务构件名称	优化服务构件名称
配送管理系统	1.学生下单	1.1 填写订单信息	1.1.1 配送外卖类型信息	---	SelectDeliverKind
			1.1.2 使用优惠券	UseCoupon	---
			1.1.3 填写订单	AddOrder	---
			1.1.4 填写收件地址等个人信息	Add Information	---
订单管理系统	订单管理	1.2 支付金额	1.2.1 计算配送费	Compute Payment	
			1.2.2 选择配送方式	---	SelectPayment Method
			1.2.3 支付金额	Pay	
		1.2 发布订单	1.2.1 更新订单列表	UpdateOrder List	---
		1.3 修改订单	1.3.1 判断是否接单	IsTaking	---
			1.3.2 删除订单	DeleteOrder	
配送管理系统	2.配送员抢单	2.1 选择订单	2.1.1 刷新列表	RefreshOrder List	

			2.1.2 订单推荐	--	recommend Orders
			2.1.2 确认订单	SelectOrder	SelectOrders
	3.订单配送	3.1 联系客户	3.1.1 返回客户联系方式	GetCustPhone	
		3.2 确认订单送达	3.2.1 完成订单	FinishOrder	
后台管理系统	4.错误监听	4.1 监听订单信息	4.1.1 报告无效订单	---	report InvaildOrder
			4.1.2 报告恶意订单	---	report BadOrder
	5.返回错误信息	5.1 返回错误信息	5.1.1 返回错误信息	---	GetReport
	6. 撤回订单	6.1 撤回订单	6.1.1 撤回订单	---	Rollback Order
活动栏管理系统	7.发布资讯	7.1 店商上传咨询	7.1.1 编辑帖子	editPost	---
后台管理系统		7.2 审核帖子	7.2.1 内容审核	ManualCheck	TextNatural LaguageProcess
			7.2.2 报告审核信息	reportMessage	
			7.2.3 撤回帖子	RollbackPost	
活动栏管理系统	8.订阅资讯	8.1 显示资讯列表	8.1.1 读取资讯	GetPosts	
		8.2 资讯推荐	8.2.1 统计用户偏好	---	Find UserReference
			8.2.2		
			8.2.3 根据偏好	---	SortPost with

			排序资讯		Reference
	9 发布优惠券	9.1 优惠券生成	9.1.1 设定期限和使用范围	CreateCoupon	

#相关地图 API 代码

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
    <style type="text/css">
        body, html{width: 100%;height: 100%;margin:0;font-family:"微软雅黑";font-size:14px;}
        #l-map{height:300px;width:100%;}
        #r-result{width:100%;}
    </style>
    <script type="text/javascript" src="//api.map.baidu.com/api?v=2.0&ak=您的密钥"></script>
    <title>关键字输入提示词条</title>
</head>
<body>
    <div id="l-map"></div>
    <div id="r-result">请输入:<input type="text" id="suggestId" size="20" value="百度" style="width:150px;"
/></div>
    <div id="searchResultPanel" style="border:1px solid #C0C0C0;width:150px;height:auto;
display:none;"></div>
</body>
</html>
<script type="text/javascript">
    // 百度地图 API 功能
    function G(id) {
        return document.getElementById(id);
    }

    var map = new BMap.Map("l-map");
    map.centerAndZoom("北京",12);                    // 初始化地图,设置城市和地图级别。
```

```
var ac = new BMap.Autocomplete(    //建立一个自动完成的对象
    {"input" : "suggestId"
    ,"location" : map
});

ac.addEventListener("onhighlight", function(e) {  //鼠标放在下拉列表上的事件
var str = "";
    var _value = e.fromitem.value;
    var value = "";
    if (e.fromitem.index > -1) {
        value = _value.province +  _value.city +  _value.district +  _value.street +  _value.business;
    }
    str = "FromItem<br />index = " + e.fromitem.index + "<br />value = " + value;

    value = "";
    if (e.toitem.index > -1) {
        _value = e.toitem.value;
        value = _value.province +  _value.city +  _value.district +  _value.street +  _value.business;
    }
    str += "<br />ToItem<br />index = " + e.toitem.index + "<br />value = " + value;
    G("searchResultPanel").innerHTML = str;
});

var myValue;
ac.addEventListener("onconfirm", function(e) {    //鼠标点击下拉列表后的事件
var _value = e.item.value;
    myValue = _value.province +  _value.city +  _value.district +  _value.street +  _value.business;
    G("searchResultPanel").innerHTML ="onconfirm<br />index = " + e.item.index + "<br />myValue = " +
myValue;

    setPlace();
});

function setPlace(){
    map.clearOverlays();    //清除地图上所有覆盖物
    function myFun(){
```

```
            var pp = local.getResults().getPoi(0).point;    //获取第一个智能搜索的结果
            map.centerAndZoom(pp, 18);
            map.addOverlay(new BMap.Marker(pp));    //添加标注
        }
        var local = new BMap.LocalSearch(map, { //智能搜索
          onSearchComplete: myFun
        });
        local.search(myValue);
    }
</script>
```

#路径规划代码：

```
#include <iostream>
#include <vector>
#include <stdio.h>
#include <math.h>
using namespace std;

int k = 30;         //引力的增益系数
int m = 15;         //斥力的增益系数
float threshold = 5; //障碍物影响距离，超过此距离斥力为 0
float S = 0.2;      //步长            与速度有关
int STEP = 200;     //循环迭代次数     与速度距离有关  后续需要根据距离来计算

struct Data
{
    int vn; // Vertex number;
    int hn; // Hazard number;

    vector<float> vx;
    vector<float> vy;
    vector<float> hx;
    vector<float> hy;

    vector<vector<float>> vdis;
};
```

```
int Sign(float num)
{
    return (num > 0) - (num < 0);
}

double Distance(double x, double x1, double y, double y1)
{
    return sqrt(pow(y - y1, 2) + pow(x - x1, 2));
}

void ComputeTraction(float ox, float oy, float xend, float yend, double angle, double *F)
{
    F[0] = k * sqrt(pow(yend - oy, 2) + pow(ox - xend, 2)) * cos(angle); // x
    F[1] = k * sqrt(pow(yend - oy, 2) + pow(ox - xend, 2)) * sin(angle); // y

    //    printf("F[0] = %lf, F[1] = %lf \n ", F[0], F[1]);
}
void ComputePotential(double angle, double theta[], double OtoEnd_Dis, double OtoHazard_Dis[], double
EtoHazard[], int n, double *PF)
{

    double Yrerx[n], Yrery[n], Yatax[n], Yatay[n];
    for (int i = 0; i < n; i++)
    {
        Yrerx[i] = 0;
        Yrery[i] = 0;
        Yatax[i] = 0;
        Yatay[i] = 0;
    }
    PF[0] = 0;
    PF[1] = 0;
    double temp = 0;

    for (int i = 0; i < n; i++)
    {
        if (OtoHazard_Dis[i] > threshold)
        {
```

```
            Yrerx[i] = 0;
            Yrery[i] = 0;
            Yatax[i] = 0;
            Yatay[i] = 0;
        }
        else if (OtoHazard_Dis[i] >= threshold / 2)
        {
            temp = m * (1 / OtoHazard_Dis[i] - 1 / threshold) / pow(OtoHazard_Dis[i], 2) * pow(OtoEnd_Dis, 2);
            Yrery[i] = -temp * sin(theta[i]);
            Yrerx[i] = -temp * cos(theta[i]);

            Yatax[i] = m * pow((1 / OtoHazard_Dis[i] - 1 / threshold), 2) * OtoEnd_Dis * cos(angle);
            Yatay[i] = m * pow((1 / OtoHazard_Dis[i] - 1 / threshold), 2) * OtoEnd_Dis * sin(angle);
        }
        else if (OtoHazard_Dis[i] > 0)
        {
            temp = m * (1 / OtoHazard_Dis[i] - 1 / threshold) * sqrt(OtoEnd_Dis) / OtoHazard_Dis[i];
            Yrerx[i] = -temp * cos(theta[i]);
            Yrery[i] = -temp * sin(theta[i]);
            Yatay[i] = 0.5 * m * pow((1 / OtoHazard_Dis[i] - 1 / threshold), 2) * sqrt(OtoEnd_Dis) * sin(angle);
            Yatax[i] = 0.5 * m * pow((1 / OtoHazard_Dis[i] - 1 / threshold), 2) * sqrt(OtoEnd_Dis) * cos(angle);
        }
        for (int i = 0; i < n; i++)
        {
            PF[1] += Yrery[i] + Yatay[i];
            PF[0] += Yatax[i] + Yatax[i];
        }
    }
}
void ComputeAngle(vector<float> hx, vector<float> hy, int n, float x, float y, double *theta)
{
    for (int i = 0; i < n; i++)
    {
        float temp = pow(hx[i] - x, 2) + pow(hy[i] - y, 2);
        //        printf("%f",(hx[i] - x) / sqrt(temp));
        theta[i] = Sign(hy[i] - y) * acosf((hx[i] - x) / sqrt(temp));
    }
```

```
}
void Update(vector<vector<float>> vdis, float *dis, int vn, int k, int used[], int *parent)
{

    for (int i = 0; i < vn; i++)
        if (!used[i] && vdis[k][i] + dis[k] < dis[i])
        {
            dis[i] = vdis[k][i] + dis[k];
            parent[i] = k;
        }
}

int FindNext(float *dis, int vn, int *used) //, int k, int *parent
{
    int min = 0;
    for (int i = 0; i < vn; i++)
        if (!used[i] && dis[i] < dis[min])
            min = i;
    //    parent[min] = k; 测试
    used[min] = 1;

    return min;
}

void Global(int vn, vector<vector<float>> vdis, int start, int *parent)
{
    int used[vn];
    used[start] = 1;
    for (int i = 0; i < vn; i++)
        used[i] = 0;
    used[start] = 1;
    float dis[vn];
    int k = start;
    for (int i = 0; i < vn; i++)
        dis[i] = vdis[k][i]; // preprocess; .. distance from k to i
    for (int j = 0; j < vn; j++)
        printf("%f ", dis[j]);
```

```
    printf("\n");
    for (int i = 0; i < vn - 1; i++)
    {
        k = FindNext(dis, vn, used);
        printf("%d.", k);
        Update(vdis, dis, vn, k, used, parent);
        for (int j = 0; j < vn; j++)
            printf("%f ", dis[j]);
        printf("\n");
    }
    for (int j = 0; j < vn; j++)
        printf("(%d,parent[%d]),", j, parent[j]);
    printf("\n");
}

void Local(vector<float> hx, vector<float> hy, int hn, float ox, float oy, float xend, float yend, vector<float>
*xroute, vector<float> *yroute)
{
    int num = 0;
    int flag = 0;
    int min = 0;
    for (int i = 0; i < STEP; i++)
    {
        num++;

        double theta[hx.size()]; // theta is the angle from current set to hazard.

        double TF[2], PF[2];
        double OtoEnd_Dis = Distance(ox, xend, oy, yend);
        double OtoHazard_Dis[hx.size()];
        double EtoHazard_Dis[hx.size()];
        for (int i = 0; i < hx.size(); i++)
        {
            OtoHazard_Dis[i] = Distance(ox, hx[i], oy, hy[i]);
            EtoHazard_Dis[i] = Distance(xend, hx[i], yend, hy[i]);
        }
        double angle = Sign(yend - oy) * acos((xend - ox) / OtoEnd_Dis);
```

```
        ComputeAngle(hx, hy, hn, ox, oy, theta);

        ComputeTraction(ox, oy, xend, yend, angle, TF);
        ComputePotential(angle, theta, OtoEnd_Dis, OtoHazard_Dis, EtoHazard_Dis, hn, PF);
        min = OtoHazard_Dis[0];
        for (int i = 0; i < hn; i++)
            if (OtoHazard_Dis[min] > OtoHazard_Dis[i])
                min = i;

        if (abs((yend - hy[min]) * (hy[min] - oy) - (xend - hx[min]) * (hx[min] - ox)) < 1e-1)
        {
            PF[0] = PF[1] + 2 * PF[0];
            PF[1] = PF[1] - 2 * PF[0];
        }
        float temp = (TF[1] + PF[1]) / (TF[0] + PF[0]);

        //        printf("angle = %lf, TF[1] =  %f TF[0] = %f, PF[1] = %f, PF[0]= %f, temp = %f\n",angle,TF[1],TF[0],
PF[1], PF[0], atan(temp));
        //        printf("temp = %f,%f,%f\n", temp,atan(temp), cos(atan(temp)));
        ox = ox + Sign(TF[0] + PF[0]) * S * abs(cos(atan(temp)));
        oy = oy + Sign(TF[1] + PF[1]) * S * abs(sin(atan(temp)));

        xroute->push_back(ox);
        yroute->push_back(oy);

        if (Distance(ox, xend, oy, yend) < 0.1)
        {
            printf("(ox,oy =%f,%f ,xend=%f,%f, %f)", ox, oy, xend, yend, Distance(ox, xend, oy, yend));
            break;
        }
    }
    printf("\n %d \n", num);
    //    printf("ans :\n");
    //    for(int i = 0; i < STEP && xtemp[i] >= 0; i++)printf("%f,", xtemp[i]);
    //    printf("\n");
    //    printf("\n");
```

```
    //    for(int i = 0; i < STEP && xtemp[i] >= 0; i++)printf("%f,", ytemp[i]);
    //    printf("\n");
    //    printf("\n");
}
Data Create()
{
    Data q;
    vector<float> temp;

    // should take random.
    q.hx = {2, 3, 4, 3, 6, 5.5, 8};
    q.hy = {1.2, 2.5, 4.5, 6, 2, 5.5, 8.5};
    q.vx = {0, 1, 4, 6, 2, 5, 8.4, 7, 10};
    q.vy = {0, 1.2, 1.4, 6.5, 5, 8, 4.5, 8.5, 10};
    //    q.start[0] = 0;q.start[1] = 0;
    //    q.end[0] = 10;q.end[0] = 10;
    q.vn = 9;
    q.hn = 7;
    for (int i = 0; i < q.vn; i++)
    {
        float temp1 = 1000000;
        for (int j = 0; j < q.vn; j++)
            if (i != j)
            {
                temp1 = Distance(q.vx[i], q.vx[j], q.vy[i], q.vy[j]);
                if (temp1 < 7)
                    temp.push_back(temp1);
                else
                    temp.push_back(1000000);
            }
            else
                temp.push_back(1000000);
        q.vdis.push_back(temp);
        temp.clear();
    }
    return q;
}
```

```
int main(int argc, const char *argv[])
{
    struct Data q = Create();
    int parent[q.vn];
    for (int i = 0; i < q.vn; i++)
        parent[i] = i;

    for (int i = 0; i < q.vn; i++)
    {
        for (int j = 0; j < q.vn; j++)
            cout << q.vdis[i][j] << ",";
        cout << "\n"
             << endl;
    }
    cout << "\n"
         << endl;
    for (int i = 0; i < q.vn; i++)
        parent[i] = 0;
    Global(q.vn, q.vdis, 0, parent);
    cout << "\n"
         << endl;
    vector<float> xroute, yroute;
    int a[7] = {17, 14, 29, 11, 200, 11, 200}, j = 1;

    for (int i = q.vn - 1; parent[i] != i; i = parent[i])
    {
        printf(" i = %d, parent,son(%.2f,%.2f), ", i, q.vx[parent[i]], q.vx[i]);
        Local(q.hx, q.hy, q.hn, q.vx[parent[i]], q.vy[parent[i]], q.vx[i], q.vy[i], &xroute, &yroute);
    }
    for (int i = 0; i < xroute.size(); i++)
    {
        if (j == 70)
            printf("\n\n");
        else
            printf("%.2f ", xroute[i]);
    }
    printf("\n\n");
```

```
    j = 1;
    for (int i = 0; i < xroute.size(); i++)
    {
        if (j == 70)
            printf("\n\n");
        else
            printf("%.2f ", yroute[i]);
    }
}
```