## 1 模型

### 1.1 架构

道路一侧部署了覆盖半径为 的BS，BS中配置了VEC服务器，服务器的计算能力用  表示。用集合表示某一时间段中BS覆盖范围内的所有产生任务的车辆，每个车辆一次只产生一个计算任务。

假设以BS覆盖范围的中心为坐标原点，道路为轴，车辆的行驶方向（向右）为正方向。车辆以恒定的速度行驶，方向固定向右行驶，在当前通信范围内的坐标为。任务车辆所产生的任务由三元组表示。其中，为任务输入数据的大小，为任务所需要的计算资源量，为任务的最大时延限制。从服务器的角度出发，研究所有被卸载到服务器上的任务的执行顺序。

任务的优先级由三种影响因素综合计算得出，分别是任务最大时延限制、任务所需资源量以及任务重要性。为了消除数据单位带来的影响，对初始数据进行标准化处理。采用离差标准化，分别对这三种影响因素的原始数据进行线性变换，使其结果映射到0和1之间，再根据各自的权重计算优先级。以表示任务的第个优先级影响因素，则处理后的数据表示为：

其中，；，分别代表、以及；和分别是影响因素中所有任务数据的最大值和最小值。

对任务原始数据进行标准化处理后，依据影响因素各自的权重值计算综合优先级，公式表示如下：

其中，、、分别是、、的权重值，且满足。对于优先级相同的多个任务，按照的值从小到大依次排序。将任务存入优先队列（利用数据结构“堆”来实现，数值越小优先级越高，因此用的是“最小堆→父节点的数值小于子节点的数值”）。

### 1.2 时间

车载任务需要在严格时延限制内完成，计算任务的总完成时间。只需考虑任务的等待时间以及计算时间，不考虑任务的上传时间和结果传输时间。在计算任务等待时间时，将任务分成两种类型，一种是已经开始执行的任务，另一种是还未开始执行的任务。则等待时间计算公式如下：

对于已经开始执行的任务，等待时间为任务开始执行的时刻与到达的时刻(不就都是0时刻吗？？)的数值之差；对于还未开始执行的任务，已等待时间为当前时刻与到达时刻的数值之差。

而且，在研究执行顺序时采用可抢占式调度，可能会有部分任务因其它任务的抢占而被迫中断执行。因此，在计算任务的等待时间和执行时间时，将任务分为两种类型：正常执行的任务以及被抢占过的任务。则任务的计算时间计算公式如下：

对于正常执行的任务，它的总计算时间为任务所需的计算资源与服务器计算能力之比；对于被抢占过的任务，总计算时间为自己的计算时间加上中断的时间，而中断的时间恰好是抢占任务的计算时间。为了评估任务失败率，需要比较任务在服务器中的总完成时间与最大时延限制，令表示任务的总完成时间，是等待时间与计算时间之和。

### 1.3 任务紧迫性判定

利用车辆的坐标计算出时刻车辆与通信范围边界的距离，计算公式如下：

若车辆已驶出当前BS的覆盖范围，即 ，则中断该车辆任务的执行，将未完成的任务视为失败。车辆在覆盖范围内的剩余时间是通过车辆与范围边界的直线距离和其行驶速度计算得来，公式如下：

判断任务的优先级是否需要调整应遵循以下两点。

1. 车辆在当前通信范围内的剩余时间应满足任务的执行时间要求。

表示满足时间要求。

1. 车载任务的状态应为就绪态。

表示处于就绪态.

当任务的已等待时间超过设定的阈值（取值可以为的一半，取值不固定，可根据实验结果进行调整），且在最大时延限制内任务剩余的时间能够满足任务的计算时间，可以调高优先级；。

令表示任务较为紧急且需要调整优先级，否则。只考虑两个任务之间的抢占行为，不考虑三个任务以上叠加的抢占行为。

用表示任务的执行状态对优先级调整必要性的影响：

用作为当前是否需要调整车载任务优先级的评判标准

### 1.4 问题

为了更精确地计算服务器中任务的失败率，选取某一时间段内的任务，将任务总数记为。在满足任务时延约束下以最小化任务失败率为目标。计算成功完成的任务数量。

最小化任务失败率的目标：

其中，第一个约束表示优先级的三个影响因素的权重值之和为1；第二个约束表示任务的总完成时间要在其最大时延限制内；第三个约束表示生成的车辆坐标要保持在服务器通讯范围内；第四个约束表示任务等待时间的判定阈值不应超过最大时延限制，否则任务按时完成的可能性就为0。

## 2 抢占式调度

### 2.1 动态调整任务优先级

利用公式（2）计算，将任务存入优先队列（利用数据结构“堆”来实现，数值越小优先级越高，因此用的是“最小堆→父节点的数值小于子节点的数值”）。若同时有多个任务的初始优先级相同，再继续考虑任务的，将小的任务排在前面。然后，服务器每隔一段时间（这个时间从[3,10]s中选取，选择结果最好的那个秒数即可，如若选好请帮我标注以下）就去检测优先队列中有无的任务，如果有，则调高对应任务的优先级。直接将紧急任务的优先级调整为最高，即优先队列中所有任务的优先级的最小值，方便任务对处于运行态的任务进行抢占。

### 2.2 抢占式调度

任务的最大可等待时间是任务最大时延限制与总完成时间之间的差值，用来表示任务在最大时延限制内可以被耽搁的最长时间。公式表示为：

将任务的最大可等待时间与任务优先级同时作为抢占行为的判断条件，假设当前执行的任务是A，要抢占的任务是B。

所有任务都完成执行后，输出任务的执行顺序。

**算法**

1. 初始化，，，，，，，；
2. for i = 1 to do
3. ，存入队列；
4. 将任务按照优先级存入优先队列中，如果优先级

数值相同，则按照的值从小到大排列；

1. while 队列不为空:
2. 每隔几秒去检测：
3. if = 1:
4. if ：
5. if ：
6. ；
7. 允许任务抢占；
8. else：
9. 不用抢占，执行任务A；
10. else：
11. 无需抢占；
12. end if
13. end for

## 3 实验

### 3.1 仿真参数设置

使用Python进行实验，在不失一般性的前提下，考虑如下场景：取一段长为的直线郊区道路，道路上的车辆速度为。在道路一侧的正中央部署了一个覆盖半径为的基站，基站上安装了一个轻量级的VEC服务器，为行驶中的车辆提供计算服务,计算能力的最大值为，每辆任务车产生一个独立任务，任务的数据大小，所需的计算资源，最大时延限制。车辆与BS之间的链路带宽，信道增益，车辆的传输功率，噪声功率。任务重要性因子取值为[1,2,3]，取值越小，越重要。

### 5.2 仿真实验

用Python进行仿真实验，并引入以下方案与本文算法进行数值比较。

（1）静态优先级调度方案。服务器根据任务的初始优先级进行排序，不考虑处理过程中任务优先级的变化。

（2）动态最高优先数优先调度（HRRN）方案。在此算法中，优先级会随着等待时间的变化而变化，优先级的计算公式为：。等待时间较长的任务优先级会更高，可以更快地得到执行。

（3）直接抢占调度。当有优先级更高的任务到达时，直接抢占，不考虑当前任务的执行状态。

本文的目标为最小化任务失败率。需要画出的对比图有：

（图1图2图3图5为折线图，图4为柱状图）

~~另外，在出结果图之前，需要先有一个输入界面（输入内容可以是最大的任务数量、任务最大时延限制，因此这个输入值会影响到结果图里面自变量的最大值部分）~~

图1：对比本文算法和（1）（2），自变量是服务器的计算能力（取值为5-10GHz），因变量是任务失败率

图2：对比本文算法和（1）（2），自变量是任务最大时延限制（取值为0-3s），因变量是任务失败率

图3：对比本文算法和（1）（2），自变量是任务数量（取值为30-100），因变量是任务失败率

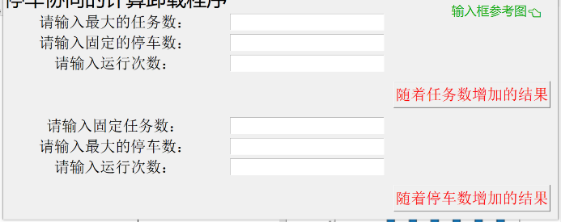
图4：对比本文算法和算法（3），自变量是任务数量（取值为30-100），因变量是任务切换（抢占）次数。

图5：只需运行本文算法，自变量为不同的检测时间间隔，因变量为任务失败率。

对于自变量为任务数量的实验结果，若失败率或切换次数的变化幅度不够大，可以适当增加任务数量，以便获得清晰的结果曲线。

对于检测时间间隔，只有图5是变化的，图1-图4的检测时间间隔是固定的，选取最好的即可。

输入界面示例图：



最后，如果本文的算法结果不如对比算法，可以适当调整参数令本算法的结果最好！！！