2 系统模型与问题形成

在移动边缘计算环境中，智能车辆通过无线网络连接到附近基站，基站将车辆的数据请求卸载到附近的边缘服务器进行计算与处理。边缘服务器部署的系统模型如图 1 所示，主要可分为两层，分别为边缘层和终端层。边缘层主要由基站和边缘服务器组成，终端层由智能汽车组成。假设考察一个矩形路面区域，矩形区域的长和

宽分别表示为 $x_{\max}$ 与 $y_{\max}$，在该区域内共布设有 I 个基站，基站集合可表示为 $\mathbf{B}=[b_1,\cdots,b_i,\cdots,b_I]$。在这样的场景下开展边缘服务器的部署，优化目标是最小化边缘服务器部署数量，兼顾最小化基站与边缘服务器的通信距离与卸载均衡之间的平衡，该优化问题 P1 可表示为公式（1）

$$
\begin{aligned}
P1 \quad & \min \quad \alpha \cdot \underbrace{\bar{D}}_{\substack{\text{基站与关联边缘}\\ \text{服务器的平均距离}}} + (1-\alpha)\cdot \underbrace{\bar{M}}_{\substack{\text{各边缘服务器}\\ \text{的平均均衡度}}} \\
& \min \quad J \\
s.t. \quad & u_{ij} \in \{0,1\},\ j = 1 \cdots J \\
& d_{ij} \le d_{\max} \\
& x_i, x_j \le [0, x_{\max}], i = 1 \cdots I,\ j = 1 \cdots J \\
& y_i, y_j \le [0, y_{\max}], i = 1 \cdots I,\ j = 1 \cdots J
\end{aligned}
\tag{1}
$$

其中，$\bar{D}$ 代表基站与关联边缘服务器的平均距离，$\bar{M}$ 代表各边缘服务器的平均卸载均衡度，α 为平衡卸载距离与卸载均衡之间的权值，J 为边缘服务器数量。优化问题 P1 的优化目标有两个，一是区域全覆盖下最小化边缘服务器的部署数量，二是兼顾卸载距离与卸载均衡的联合最小化。

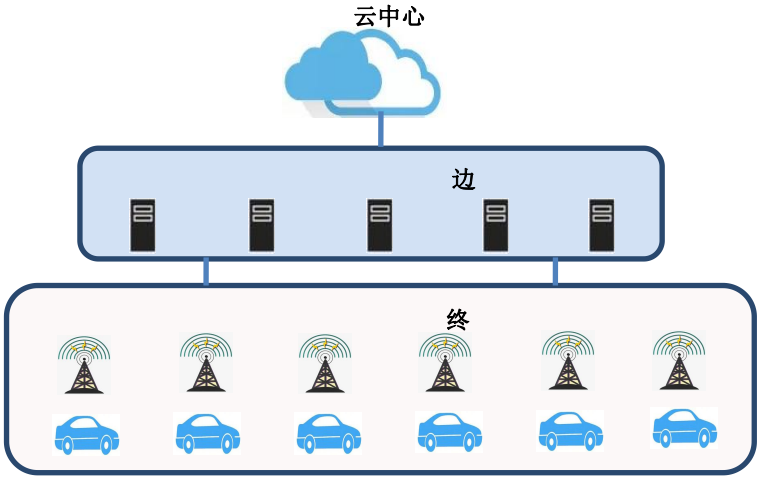


图 1　系统模型图

优化问题 P1 是多目标优化的 NP 难问题，本文将该优化问题的两个优化目标转化为两个子优化问题，假设基站已经携带智能车辆的数据请求，需要解决的两个子问题是：1）边缘服务器的部署位置与部署数量，实现基站全覆盖下的以就近卸载为主要目标的最小数量边缘服务器部署，降低传输时延和部署成本；2）优化基站与边缘服务器的匹配关系，兼顾就近卸载与卸载均衡的平衡，有效降低运营商的计算资源配置成本，同时保障卸载质量。

3　基于最短距离密度的边缘服务器部署方法

由于在所考察的路面区域内，存在任意位置设置通信设备并不现实，安装位置具有局限性，因此，以基站

位置为边缘服务器的布局基准位置实施部署，即边缘服务器的部署位置是在已部署的基站位置中确定。本文基于该条件，提出了基于最短距离密度的区域全覆盖部署方法，方法的具体流程为：

第一步：计算任一基站b_i与其它基站的距离，将处在基站 b_i覆盖范围内的基站集合按照距离大小进行升序排序，距离越近，排序位置越靠前，可表示为公式（1）

$$\mathbf{b}_{index} = [\mathbf{b}^{1}_{index} \cdots \mathbf{b}^{i}_{index} \cdots \mathbf{b}^{I}_{index}]^{T} \tag{1}$$

其中，$\mathbf{b}^{i}_{index} = [x^{i}_{1} \cdots x^{i}_{v} \cdots x^{i}_{V}], i \in 1 \cdots I$ 表示基站b_i覆盖范围内的基站集合，V 表示基站 b_i覆盖范围内的基站数量，x^{i}_{v}表示基站 b_i覆盖范围内距离按照升序排

列中的第 v 个基站索引，x^{i}_{1}表示基站 b_i覆盖范围内距离最近的基站索引。

第二步：根据各基站$\mathbf{b}^{i}_{index}$的首位元素$x^{1}_{v}, i \in 1 \cdots I$（距离各基站最近的基站索引）所对应的相同索引出现次数，得到多个最短距离密度，可表示为公式（2）

$$\begin{cases} x^{i_{m1}}_{1} = x^{i_{mm}}_{1} = \cdots = x^{i_{mM}}_{1}, \ i_{mm} \in 1 \cdots I \\ x^{i_{n1}}_{1} = x^{i_{nn}}_{1} = \cdots = x^{i_{nN}}_{1}, \ i_{nn} \in 1 \cdots I \\ \quad \vdots \\ x^{i_{z1}}_{1} = x^{i_{zz}}_{1} = \cdots = x^{i_{zZ}}_{1}, \ i_{zz} \in 1 \cdots I \end{cases} \tag{2}$$

其中，$x^{i_{m1}}_{1} \neq x^{i_{n1}}_{1} \neq \cdots \neq x^{i_{z1}}_{1}$， $M, N, \cdots, Z$ 分别表示索引 $x^{i_{m1}}_{1}, x^{i_{n1}}_{1}, \cdots, x^{i_{z1}}_{1}$ 出现的次数

第三步：找出其中出现次数最多的索引值，可表示为 G，按公式（3）计算

$$G = max(M，\ N, \cdots, Z) \tag{3}$$

其中，G 为最短距离密度，则公式（2）中第 g 行第一个元素$x^{i_{g1}}_{1}$记录的索引所在的基站位置即为边缘服务器的部署位置，部署的边缘服务器可记为 $e_{\tilde{b}}$。同时，从基站集合$\mathbf{B}$中和向量$\mathbf{b}_{index}$中删除公式（2）中第 g 行关联的所有 G 个基站，这些基站都是边缘服务器 $e_{\tilde{b}}$的关联基站。

第四步：判断基站集合$\mathbf{B}$是否为空集，若为空集，算法停止；反之，返回步骤（2），对基站集合中剩余基站展开边缘服务器的部署计算。

第五步：得到边缘服务器的部署位置集合$\mathcal{L}_e = [L_{e_1}, \cdots, L_{e_j}, \cdots, L_{e_J}]$与部署数量 J。

算法 1 基于最短距离密度的边缘服务器部署方法

输入：基站位置（x_i，y_i）, i，$j \in [1, I]$

输出：$\mathcal{L}_e=[L_{e_1},\cdots,L_{e_j},\cdots,L_{e_J}]$

1.得到基站 i 与其他各基站的距离向量$\mathbf{d}_i, i=1\cdots I$

2.for $j=i\sim I$

*3.*计算 d_{ij}

4.end

5.end

6.根据覆盖范围阈值 $d_{\max}$ 获取 $\mathbf{b}^i_{index}=[x^i_1\cdots x^i_v\cdots x^i_V], i\in 1\cdots I$

7.如果基站集合 $\mathbf{B}\neq\varnothing$,按公式（2）进行计算，得到 G

8.for i=1~L

9.if $x^i_1=x^{i_{g1}}_1$

10.从 $\mathbf{B}$ 中删除基站 b_i，同时删除 $\mathbf{b}^i_{index}$

11.end

12.end

13.得到 $\mathcal{L}_e=[L_{e_1},\cdots,L_{e_j},\cdots,L_{e_J}]$

4 基于 Q 学习思想的兼顾卸载就近与卸载均衡平衡的多智能体调度方法

算法 1 中边缘服务器部署的关键是最短距离密度，而最短距离密度越大的边缘服务器，卸载任务越重。若仍按照算法 1 结束时确定的基站与边缘服务器的匹配关系，会导致部分边缘服务器卸载任务过重，另一部分边缘服务器资源闲置浪费。而任意调度基站与边缘服务器的匹配关系，又可能造成基站与边缘服务器距离较远，服务质量不能保证的结果。

综合上述两个原因，本文提出一种基于 Q 学习思想的卸载就近与卸载均衡平衡的多智能体调度方法，该方法将每个基站看做一个智能体，每个智能体可选择通信范围内的某个边缘服务器为卸载服务器，系统对智能体做出的选择给予奖励，并通过当前奖励与过去奖励的加权计算 Q 值。多智能体通过持续学习，将每次学习的最优动作进行记录。系统状态为每个智能体与边缘服务器的匹配关系；奖励为某智能体发生一个动作后，为卸载就近与卸载均衡所做出的贡献。算法通过在多个智能体之间进行匹配均衡，直到在已具备的场景参数下达到 Q 值最优。

为每个智能体建立一个 Q 表，每个 Q 表对应一个向量$\mathbf{Q}_i=[Q_{i1}\cdots Q_{iq}]$，$q$ 为第 i 个智能体所关联的边缘服务器数量，Q_{iq} 表示第 i 个智能体选择第 q 个动作的 Q 值，即第 i 个智能体选择匹配第 q 个边缘服务器的 Q

值，可按公式（4）计算

$$Q_{iq}^{t} = r_{iq}^{t} + \beta \cdot r_{i-optimal}^{t+1} \tag{4}$$

其中，r_{iq}^{t} 代表第 t 轮第 i 个智能体选择第 q 个动作时的奖励值，$r_{i-optimal}^{t+1}$ 代表第 i 个智能体选择第 q 个动作，到达 t+1 时刻时，选择各种未来动作所能带来的最优奖励，β 代表权重。在公式（4）中有两个关键问题：一是当前奖励值 r_{iq}^{t} 如何计算；二是未来奖励中的最优值 $r_{i-optimal}^{t+1}$ 如何确定。

针对第一个问题，由于算法目标是智能体与边缘服务器之间的匹配要实现卸载均衡与卸载就近，应以各边缘服务器的卸载均衡度与各边缘服务器及其关联智能体的平均距离为衡量规则，当前奖励 r_{iq}^{t} 可按照公式（5）计算

$$r_{iq}^{t} = c_1 \cdot \underbrace{\frac{\sum_{j=1}^{J'}\sum_{i=1}^{\left|e_{j'}\right|} d_{ij}}{J'\left|e_{j'}\right|}}_{\text{平均距离}} + c_2 \cdot \underbrace{\sum_{j=1}^{J'}\left|(\tilde{M} - M_j)\right|}_{\text{均衡度}} \tag{5}$$

其中，J' 代表智能体 i 在第 t 轮选择动作 q 后，与智能体 i 匹配发生变化的两个基站-边缘服务器集合，$\left|J'\right|$ 代表 J' 集合中的基站-边缘服务器的数量，$\left|J'\right| = 2$ 。$e_{j'}$ 表示发生变化的第 j' 个边缘服务器在变化后所关联的基站集合，$\left|e_{j'}\right|$ 代表该集合数量。公式（5）中的第一部分代表平均距离的变化，只与发生变化的两个智能体-边缘服务器集合有关，而与其他智能体-边缘服务器集合无关。d_{ij} 代表第 i 个智能体与第 j 个边缘服务器之间的距离，其值越小越好。公式（5）中的第二部分代表各边缘服务器资源使用的均衡程度，M_j 为第 j 个边缘服务器使用的计算资源，$\tilde{M}$ 为所需计算资源的平均值。当各边缘服务器之间的均衡度越低，其值越大，边缘服务器闲置资源越多，成本越高；当边缘服务器之间的均衡度越高，其值越小，闲置资源越少，成本越低。c_1 与 c_2 分别表示卸载距离与卸载均衡对应的加权值，且 $c_1 + c_2 = 1$ 。综合公式（5）中的两个部分情况，奖励 r_{iq}^{t} 越小越好。

针对第二个问题，设置一个暂态最优匹配矩阵，该矩阵存储各智能体当前所有经历下的最优边缘服务器匹配关系。暂态最优向量可表示为公式（6）

$$\mathbf{x} = \begin{pmatrix} x_{1i} \\ x_{2i} \end{pmatrix}, i = 1 \cdots I \tag{6}$$

其中，$\mathbf{x}$ 是维度为 $2\times I$ 的实矩阵，x_{1i} 记录智能体 i 所匹配的当前所有经历中 Q 值最小（当前最优）的边缘服务器索引，x_{2i} 记录智能体 i 所匹配的当前最优边缘服务器所对应的 Q 值。智能体选择动作 q（即选择边缘服务器 q）时对应的未来奖励可通过查找暂态最优匹配向量并按照公式（7）进行计算

$$r_{i-\min}^{t+1}=\begin{cases}=0,\ x_{1i}=q\\ =rand(0,r_{iq}^{t}),\ x_{1i}\neq q\end{cases} \tag{7}$$

其中，$rand(0,r_{iq}^{t})$ 代表在 0 与 r_{iq}^{t} 之间的一个随机数。

未来奖励是对智能体选择一个动作后，该动作为后续发展可能带来的效益的体现。与奖励 r_{iq}^{t} 的变化趋势保持一致，$r_{i-optimal}^{t+1}$ 也越小越好。当智能体 i 选择匹配边缘服务器 q 时，查找暂态最优向量中智能体 i 是否记录着边缘服务器 q。如果是，代表过去所有经历中，智能体 i 选择匹配边缘服务器 q 最优，对未来实现更好收益的可能性大，所以取 $r_{i-\min}^{t+1}=0$；如果否，代表过去所有经历中，智能体 i 选择匹配边缘服务器 q 不是最优匹配，对未来实现更好收益的可能性小，所以取 $r_{i-\min}^{t+1}=rand(0,r_{iq}^{t})$。当 r_{iq}^{t} 越大，$rand(0,r_{iq}^{t})$ 可能就越大，反之，$rand(0,r_{iq}^{t})$ 可能就越小。

各智能体按照顺序，逐一进行动作选择，更新 Q 值，并更新与存储暂态最优匹配向量。当各智能体 Q 值不再变化或达到迭代次数，算法结束，暂态最优向量 $\mathbf{x}$ 中的第一行元素值就是各智能体与边缘服务器的最终匹配关系。算法 2 的具体流程如下：

算法 2 基于 Q-learning 思想的兼顾卸载就近与卸载均衡的多智能体调度方法

输入：$\mathcal{L}_e=[L_{e_1},\cdots,L_{e_j},\cdots,L_{e_J}]$

输出：各基站与边缘服务器的匹配关系

1.for k=1~K

2.for i=1~J

3.智能体 i 选择动作 q

4.计算 r_{iq}^{t}

5.计算 $r_{i-\min}^{t+1}$

6.计算 Q_{iq}^{t}

7.比较 Q_{iq}^{t} 与 x_{2i}

8.if $Q_{iq}^{t} > x_{2i}$

9.保持 x_{1i}

10.else

11. $x_{1i} = q$

12. $x_{2i} = Q_{iq}^{t}$

13.end

14.end

15.end

5 性能分析

算法 1 解决了边缘服务器的部署位置与部署数量问题，算法既照顾到每个基站的卸载需求，实现了区域全覆盖，又根据基站的最短距离密度，迭代寻找当前基站集合中密度最大的基站位置进行边缘服务器部署。这样部署的好处是：一方面保障了基站与边缘服务器之间尽可能小的卸载距离，另一方面使部署的边缘服务器尽力服务更多的附近基站，因而实现了基站与边缘服务器之间在距离尽力最短下的最小部署数量，降低了通信运营商的部署成本，保障了就近卸载的需求。

算法 2 基于 Q 学习思想，但又与传统 Q 学习算法不同。传统 Q 学习算法主要解决 Q 值不断增大的优化问题，在计算过程中在原 Q 值下不断迭代选择新动作后的奖励与未来奖励的贡献。而算法 2 中有两处与传统 Q 学习算法不相同，一是多智能体，每个智能体都有其相应的动作选择，而且每个智能体的选择会影响其他智能体的动作选择结果；二是 Q 值应越小越优秀，每个智能体每一轮 Q 值的计算不能叠加在原 Q 值之上，因为当每个智能体按需在每一轮选择动作时，系统在智能体之间的调度关系与上一轮已经发生了变化，若还在上一轮 Q 值基础上继续叠加，则与实际情况会发生较大误差。因此，所提算法 2 基于 Q 学习的强化学习思想，将当前奖励与未来奖励的加权作为 Q 值获取方法，将每个智能体在每轮动作选择时，根据最新匹配环境得到当前奖励，同时又将智能体当前得到的最优选择作为参照，对未来奖励的合理获得提供帮助。综上，算法 2 基于 Q 学习思想设计各智能体的奖励和未来最优奖励实现 Q 值的不断优化与更新同时，同时通过“更新与记忆”暂态最优智能体与边缘服务器匹配关系，使兼顾卸载就近与卸载均衡的高质量通信得到保障。

6 仿真计算

5G 基站覆盖半径约为 100-300 米，取中间值 200，设置两个计算场景：（1）在 $2000\text{m}\times 2000\text{m}$ 的矩形范围内，约应设置 25 个基站；（2）在 $4000\text{m}\times 4000\text{m}$ 的矩形范围内，约应设置 50 个基站。在两个场景下利用算法 1 进行边缘服务器部署计算，在算法 1 得到的结果下，再利用算法 2 进行边缘服务器与基站的调度匹

配，仿真参数如表 1 所示。在表 1 中，$c_1 = c_2$ 表示在进行卸载就近与卸载均衡的平衡上，二者地位同样重要；β 取值较小，可在一定程度上依赖过去经历的最优结果，但又让更多动作的选择在暂态最优向量中得以出现，很大程度避免了陷入局部最优。

表 1　仿真参数

参数	数值
边缘服务器最大覆盖范围 $d_{\max}$	800 米
基站数量	25/50
横坐标范围	0-2000 米/0-4000 米
纵坐标范围	0-2000 米/0-4000 米
c_1	0.5
c_2	0.5
β	0.3

6.1　25 个基站场景

根据算法 1，25 个基站中有 9 个基站在其位置处需要部署边缘服务器，并得到每个边缘服务器的关联基站，如表 2 所示。表 2 中第 2 行说明了要在基站索引为 2、5、10、11、13、15、18、22、25 这 9 个基站位置处设置边缘服务器；表 2 中第 3 行到第 7 行中的每列分别代表对应边缘服务器的多个关联基站；第 3 行与第 2 行相同，是因为在某个基站处如果设立边缘服务器，首先服务该位置处的基站。

表 2　边缘服务器设置及其关联基站

边缘服务器索引	1	2	3	4	5	6	7	8	9
设置边缘服务器的基站索引	2	5	10	11	13	15	18	22	25
边缘服务器关联基站 1（索引）	2	5	10	11	13	15	18	22	25
边缘服务器关联基站 2（索引）	6		1	17		3	9	4	8
边缘服务器关联基站 3（索引）	23		21	20		7	12	14	19
边缘服务器关联基站 4（索引）								16	
设置边缘服务器的基站索引								24	

边缘服务器的设置及其关联基站情况如图 2 所示。红色空心星型代表边缘服务器，蓝色空心星型代表基站，直线连接了基站与边缘服务器之间的匹配关系。

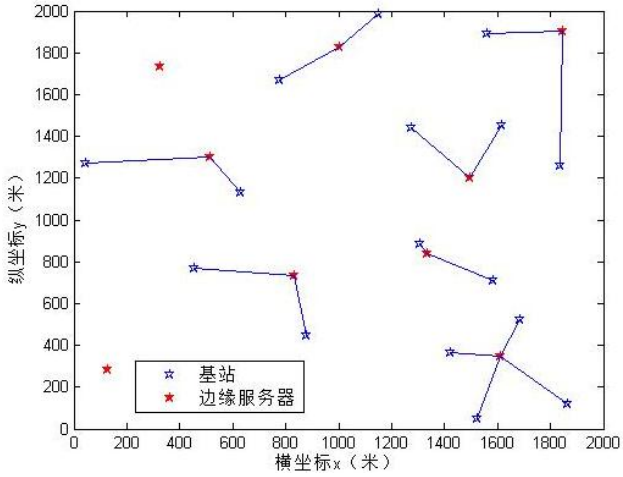
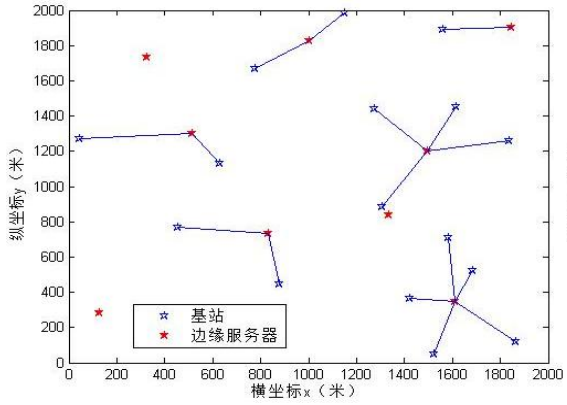
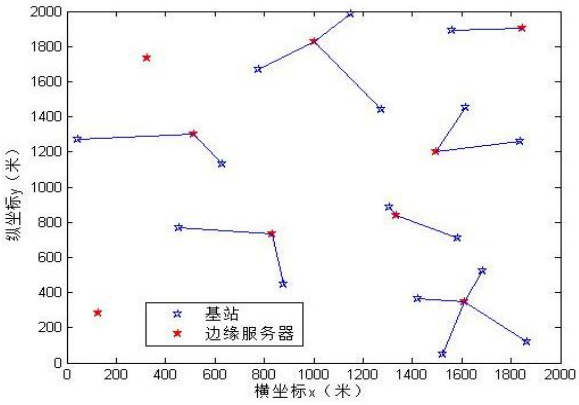


图 2 边缘服务器的设置及其关联基站情况（25 个基站）

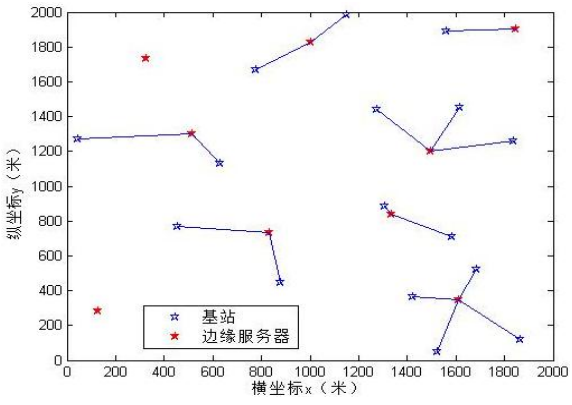
在 25 用户场景下，当边缘服务器的布局数量与位置确定，利用算法 2 进一步优化基站与边缘服务器的匹配关系。初始时，除了基站处部署边缘服务器的基站具有直接与边缘服务器的匹配关系，其他基站与边缘服务器无匹配关系。经过 600 次迭代运算，算法 2 达到收敛，如图 3 所示，得到了迭代次数为 100、500 和 600 的匹配结果。



（a）100 次迭代



（b）500 次迭代



（c）600 次迭代

图 3 边缘服务器与基站在不同迭代次数下的匹配结果（25 个基站）

从图 2 和图 3（c）可以看出，算法 2 在算法收敛时的匹配结果比算法 1 得到的匹配结果有一定程度的改善。表 3 记录了算法 2 在 100 次、500 次和 600 次迭代下的平均 Q 值。可以看出，随着迭代次数不断增加，总

Q 值与平均 Q 值不断减小，说明在算法收敛时，得到的边缘服务器与基站间匹配关系在就近卸载和卸载均衡间达到了最好的均衡。

表 3　不同迭代次数下的 Q 值

参数	平均 Q 值	总 Q 值
算法 2 匹配结果-100 次迭代（图 3（a））	141.18	2258.85
算法 2 匹配结果-500 次迭代（图 3（b））	118.75	1899.94
算法 2 匹配结果-600 次迭代（图 3（c））	118.15	1890.34