- **系统模型**
  - **联邦学习**

    给定需要解决的问题的相对精度 ϵ 和本地迭代的相对精度阈值 θ 情况下的全局迭代次数上界

    $$I^g(\epsilon,\theta)=\frac{\zeta\cdot\log\left(\frac{1}{\epsilon}\right)}{1-\theta}$$

    client 获得本地相对精度 θ_k 的本地迭代上界

    $$I_k^l(\theta_k)=\gamma_k\log\left(\frac{1}{\theta_k}\right)$$

    在给定各个 θ_k 的情况下全局迭代次数

    $$I^g(\epsilon,\theta)=\frac{\zeta\cdot\log\left(\frac{1}{\epsilon}\right)}{1-\max\theta_k}$$

  - **费用模型**
    - **communication cost**

      $$T(\theta_k)=\frac{T_k}{1-\theta_k}$$

      泰勒展开近似为

      $$T(\theta_k)=T_k(1+\theta_k)$$

    - **computing cost**

      即 $I_k^l(\theta_k)(1+\theta_k)$

    综合两种费用，得到

    $C_k(\theta_k)=(1+\theta_k)\cdot(v_k\cdot T_k+(1-v_k)\cdot\gamma_k\log\left(\frac{1}{\theta_k}\right))$

- **激励机制**

  主从博弈（Stackelberg Game）：非同时进行，后手知道先手的策略，先手知道后手知道先手的策略……双方的策略是共同知识

  - **Client(后手)**

    一开始 MEC server 会告诉 client 一个统一的收益率

    价值函数：$v_k(\theta_k)=1-\theta_k$

    client 的效用函数为

    $$u_k(r,\theta_k)=r(1-\theta_k)-C_k(\theta_k),\forall k\in\mathcal{K}$$

    因为 $u_{22}^{''}<0$，$u_k(r,\theta_k)$ 关于 θ_k 严格凹，有唯一的极大值点 $\theta_k^*(r)$

  - **MEC Server(先手)**

    令 $x(\epsilon)$ 为达到相对精确度 ϵ 的迭代次数

$$x(\epsilon) = \zeta \cdot \log\left(\frac{1}{\epsilon}\right)$$

给全局迭代次数设一个上界 δ，有

$$I^g(\epsilon, \theta) = \frac{x(\epsilon)}{1-\theta} \le \delta$$

MEC Server 的效用函数为

$$\mathcal{U}(x(\epsilon), r|\theta^*)) = \beta(1 - 10^{-(ax(\epsilon)+b)}) - r\sum_{k\in\mathcal{K}}(1-\theta_k^*(r))$$

- **引理1（KKT 条件）**

  最优化情况下 $x^*(\epsilon) = \delta(1 - \max\theta_k^*(r))$

- **Stackelberg Equilibrium**

$$\mathcal{U}(r^*, \theta^*) \ge \mathcal{U}(r, \theta^*)$$
$$u_k(\theta_k^*, r^*) \ge u_k(\theta_k, r^*)$$

  - **后手的最优策略**

$$u_k'(r, \theta_k) = 0$$
$$\log\left(e^{\frac{1}{\theta_k(r)}}\theta_k(r)\right) = \left[\frac{r + v_k T_k}{(1-v_k)\gamma_k} - 1\right] = g_k(r)$$

    最优的 $\hat{\theta}_k$ 在满足 $g_k(r) = \log\left(e^{\frac{1}{\hat{\theta}_k(r)}}\hat{\theta}_k(r)\right)$ 的时候取到

$$\theta_k^*(k) = \min\{\hat{\theta}_k, \theta_{th}\}$$

  - **先手的最优策略**

    考虑筛选 client 的策略

$$z_k = \begin{cases} 1 & \text{if } r > \hat{r}_k \\ 0 & \text{otherwise} \end{cases}$$

    其中

$$\hat{r}_k = g_k^{-1}\left(\log\left(e^{\frac{1}{\theta_{th}}}\theta_{th}\right)\right)$$

    即能让 client k 本地相对精度达到阈值 θ_{th} 的最小激励值

    所以先手需要最大化效用函数

$$\max \beta(1 - 10^{-(ax^*(\epsilon)+b)}) - r\sum_{k\in\mathcal{K}} z_k \cdot (1-\theta_k^*(r))$$

    使用如下算法

Algorithm 1 MEC Server's Utility Maximization

```
Sort clients as with r̂_1 < r̂_2 < ... < r̂_K
R = {}, A = K, j = K
while j > 0 do
    Obtain the solutions r_j to the problem
        max_{r ≥ r̂_1} β(1 − 10^{−(ax*(ε)+b)}) − r Σ_{k∈A} (1 − θ*_k(r)).
    if r_j > r̂_j, then R = R ∪ {r_j};
    end if
    A = A\j;
    j = j − 1;
end while
Return r_j ∈ R with the highest optimal values in the problem (4)
```

Sort clients as with $\hat{r}_1 < \hat{r}_2 < \ldots < \hat{r}_K$

$\mathcal{R} = \{\}, \mathcal{A} = \mathcal{K}, j = K$

while $j > 0$ **do**

Obtain the solutions r_j to the problem

$$\max_{r \geq \hat{r}_1} \beta \left(1 - 10^{-(ax^*(\epsilon)+b)}\right) - r \sum\nolimits_{k \in \mathcal{A}} (1 - \theta_k^*(r)).$$

if $r_j > \hat{r}_j$, **then** $\mathcal{R} = \mathcal{R} \cup \{r_j\}$;

end if

$\mathcal{A} = \mathcal{A} \backslash j$;

$j = j - 1$;

end while

Return $r_j \in \mathcal{R}$ with the highest optimal values in the problem (4)

最终得到的 $\{r*, \theta^*\}$ 就是 Stackelberg 均衡点