Deep Reinforcement Learning for Delay-Oriented IoT Task Scheduling in Space-Air-Ground Integrated Network

Conghao Zhou, Wen Wu, Hongli He, Peng Yang,Feng Lyu, Nan Cheng, and Xuemin (Sherman) Shen

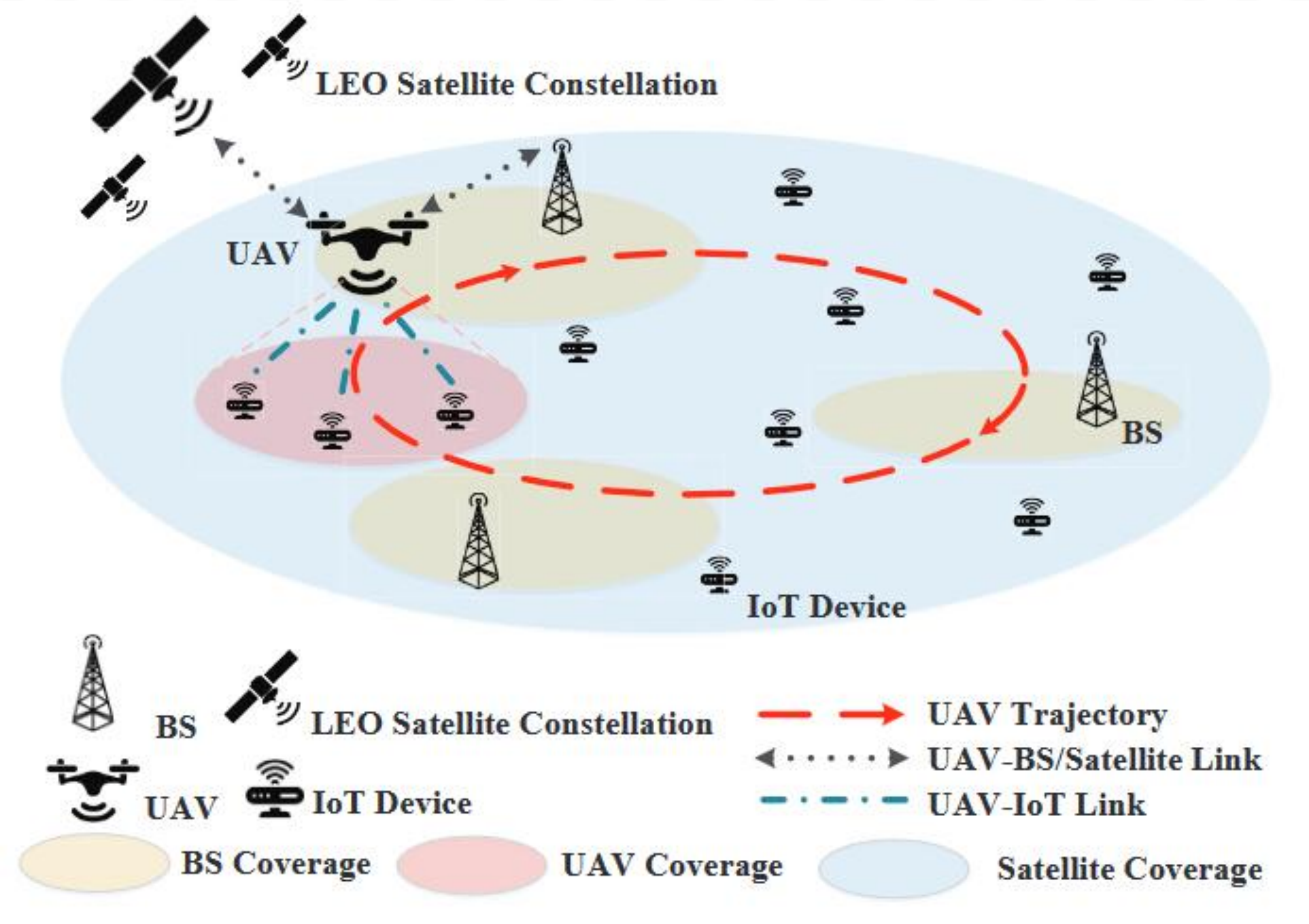
IEEE Transactions on Wireless Communications 2021

汇报人：张泷千

2023 / 11/ 1

01 INTRODUCTION

01 SAGIN Architecture



未知任务到达顺序

- 对于大量的物联网设备，任务到达是动态的，突发的，甚至是未知的，这对调度策略提出了实时性要求。

组件计算能力有差别

- 无人机，地面基站和卫星在通信和计算能力方面具有差异化特点，调度策略应该根据任务的特性选择适当的SAGIN组件进行任务处理

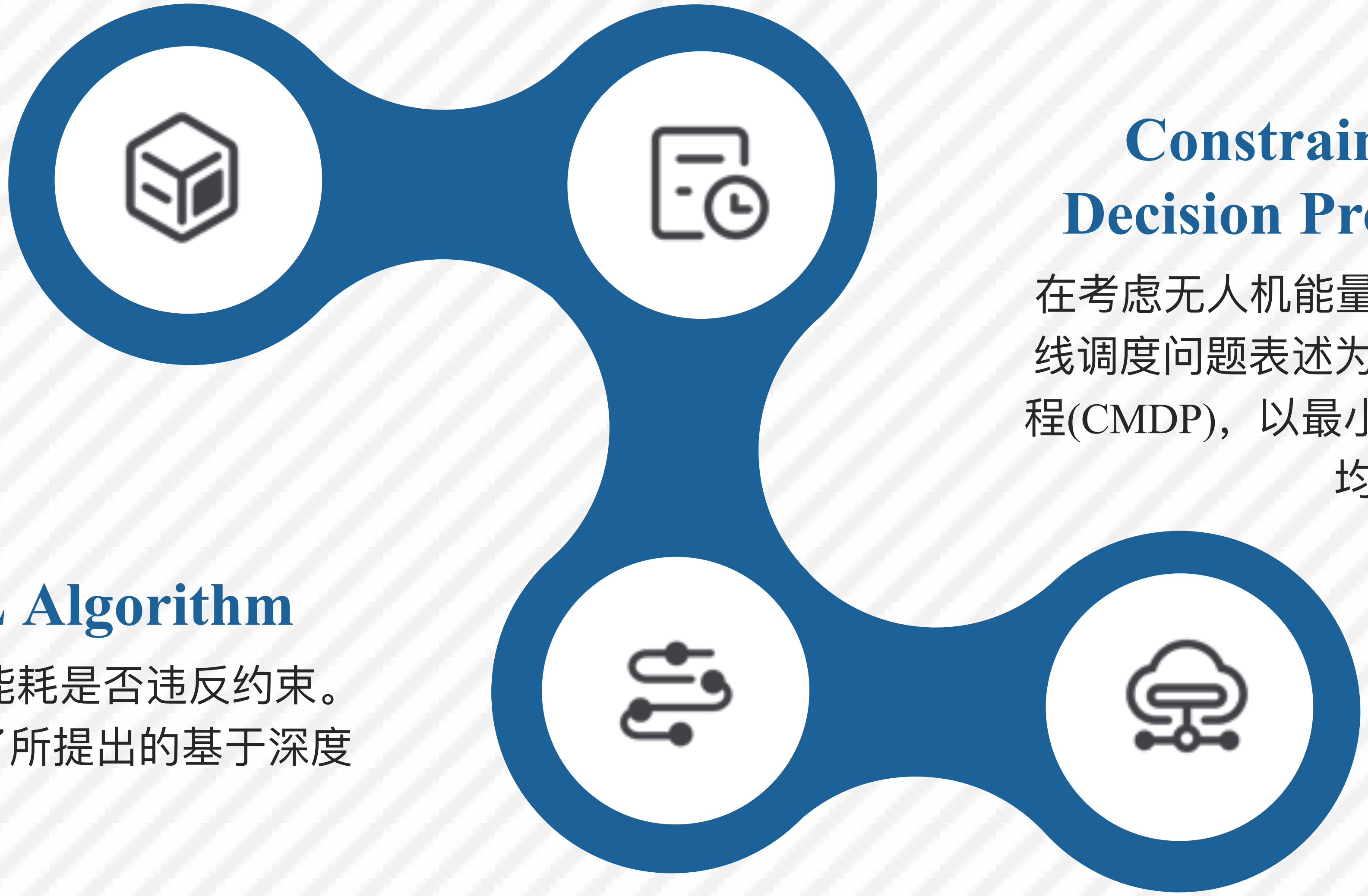
能耗要考虑两个部分

- 在调度策略中，既要考虑当前的能量消耗，也要考虑未来到达任务的能量储备

01 Contributions

Computing Task Scheduling Scheme

针对SAGIN中**面向延迟**的物联网服务，提出了一种计算任务调度方案名叫DOTS，即无人机沿轨迹飞行，收集计算任务并进行实时调度的决策



Constrained Markov Decision Process (CMDP)

在考虑无人机能量容量的情况下，将在线调度问题表述为约束马尔可夫决策过程(CMDP)，以最小化任务处理的时间平均延迟

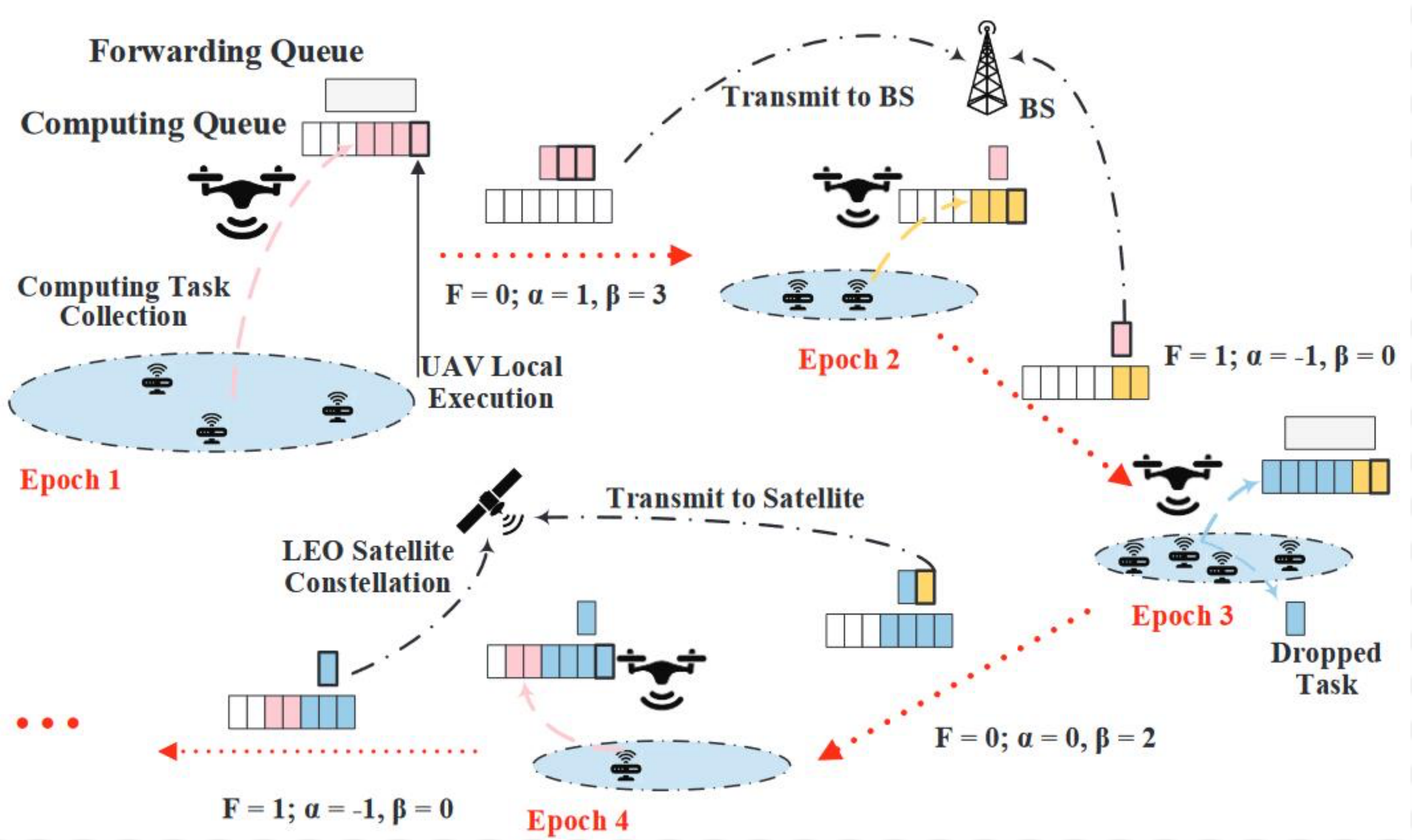
Deep Risk-Sensitive RL Algorithm

定义一个风险函数来表示无人机的能耗是否违反约束。此外利用DNN在DOTS方案中实现了所提出的基于深度强化学习的算法。

02

System Model

System Model



Computing Model

- **Task Offloading**

$$d_1(\alpha_t, \beta_t) = \frac{\beta_t \phi \gamma}{f_{\alpha_t}}, \ \alpha_t \in \mathcal{L}_t,$$

- **Local Processing**

(local computing delay&queuing delay)

$$d_2(\alpha_t, \beta_t) = \frac{\min\left\{\lfloor \frac{f_U \tau}{\phi \gamma} \rfloor, H_t\right\} \phi \gamma}{f_U} + O_t \tau,$$

Communication Model

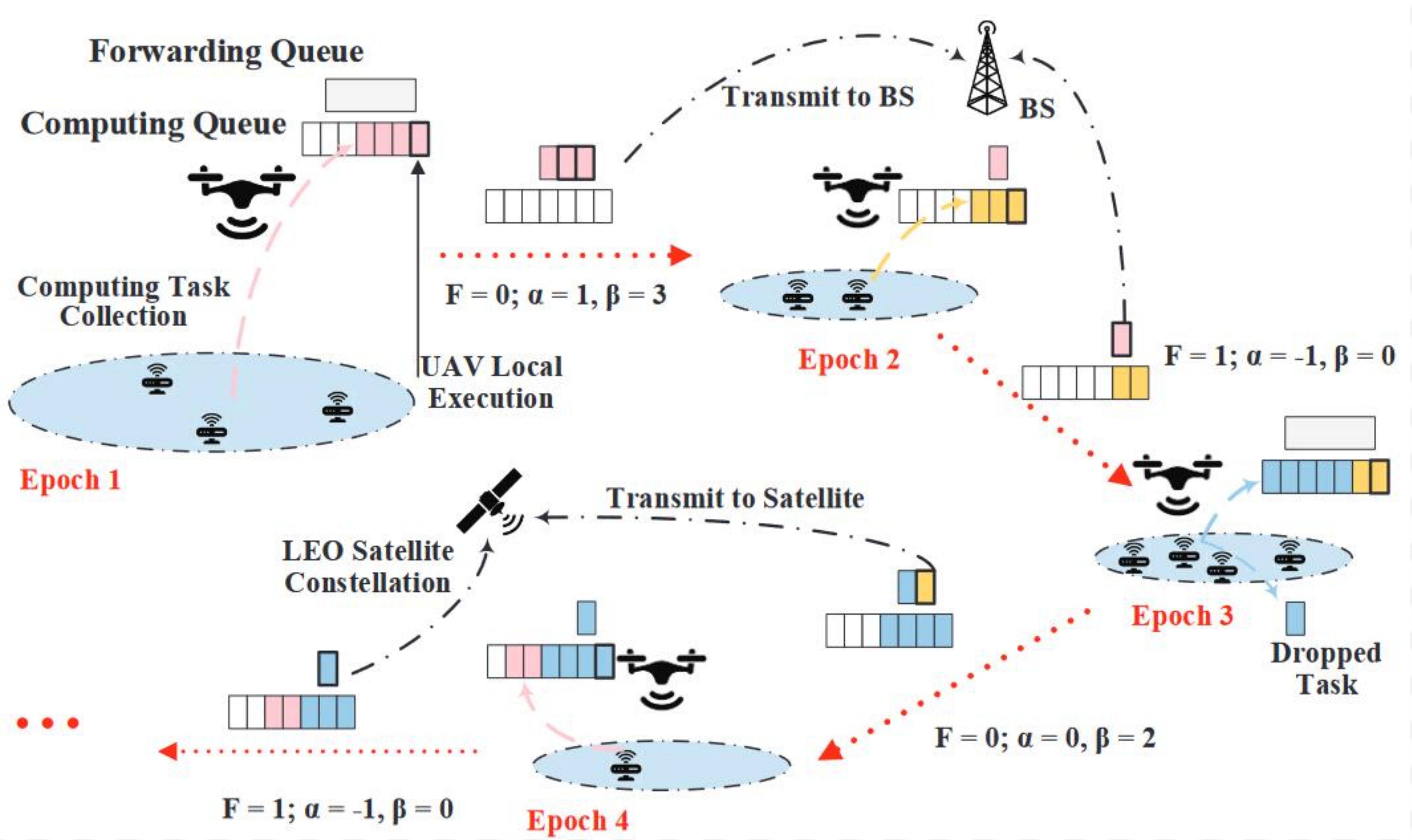
- **Offload to Satellite**

$$d_3(\alpha_t, \beta_t) = \frac{\beta_t \phi}{r_{\alpha_t}} + d_S, \quad \alpha_t = 0.$$

- **Offload to BS**

$$d_3(\alpha_t, \beta_t) = \frac{\beta_t \phi}{r_{\alpha_t}}, \quad \alpha_t \in \mathcal{L}_t, \alpha_t \neq 0,$$

System Model



Energy Consumption Model

- **Communication-related Energy**

$$e_0(\alpha_t,\beta_t)=\begin{cases}P_S d_4(\alpha_t,\beta_t), & \alpha_t=0\\ P_B d_4(\alpha_t,\beta_t), & \alpha_t\in\mathcal{L}_t,\alpha_t\neq 0\end{cases}$$

- **Computing-related Energy**

$$e_1(\alpha_t,\beta_t)=\min\{H_t\phi\gamma, f_U\tau\}\cdot\xi\,(f_U)^2.$$

Final Energy Consumption:

$$E_t=E_{t-1}+e_0(\alpha_t,\beta_t)+e_1(\alpha_t,\beta_t)$$

03

Problem Formulation

Problem Formulation

Total Delay

- 主要包含三个部分，其中卫星传输考虑到了传播延迟，三个部分分别表示之前建模阶段提到的任务卸载延迟，UAV的任务执行延迟以及通信延迟

$$D_t = \begin{cases} \frac{\beta_t \phi \gamma}{f_{\alpha_t}} + \frac{\min\left\{\lfloor \frac{f_U \tau}{\phi \gamma} \rfloor, H_t\right\} \phi \gamma}{f_U} + O_t \tau + \frac{\beta_t \phi}{r_{\alpha_t}} + d_S, & \alpha_t = 0 \\ \frac{\beta_t \phi \gamma}{f_{\alpha_t}} + \frac{\min\left\{\lfloor \frac{f_U \tau}{\phi \gamma} \rfloor, H_t\right\} \phi \gamma}{f_U} + O_t \tau + \frac{\beta_t \phi}{r_{\alpha_t}}, & \alpha_t \neq 0, \end{cases} \quad (13)$$

Constrained Markov Decision Process (CMDP)

$$M := \langle S, A, P, C, \Pi \rangle$$

- **State:** 无人机位置，发送队列里的任务，计算队列里的任务，能量消耗
- **Action:** 卸载位置以及卸载的任务数
- **State Transition:**更新后的无人机位置，发送队列里的任务，计算队列里的任务，能量消耗
- **Cost Function:** 增加与task dropping相关的系数 $C(s_t, a_t) = D_t + \Lambda_t$, $\Lambda_t = \lambda \max(M_t + O_t - \rho, 0)$
- **Policy:** 在状态s下选择a动作

04

Deep Risk-Sensitive RL Algorithm

Basic Q-Learning Algorithm

Q-Learning(Tabular Version)：

- Observe a transition (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_a Q^\star(s_{t+1}, a)$.
- TD error: $\delta_t = Q^\star(s_t, a_t) - y_t$.
- Update: $Q^\star(s_t, a_t) \leftarrow Q^\star(s_t, a_t) - \alpha \cdot \delta_t$

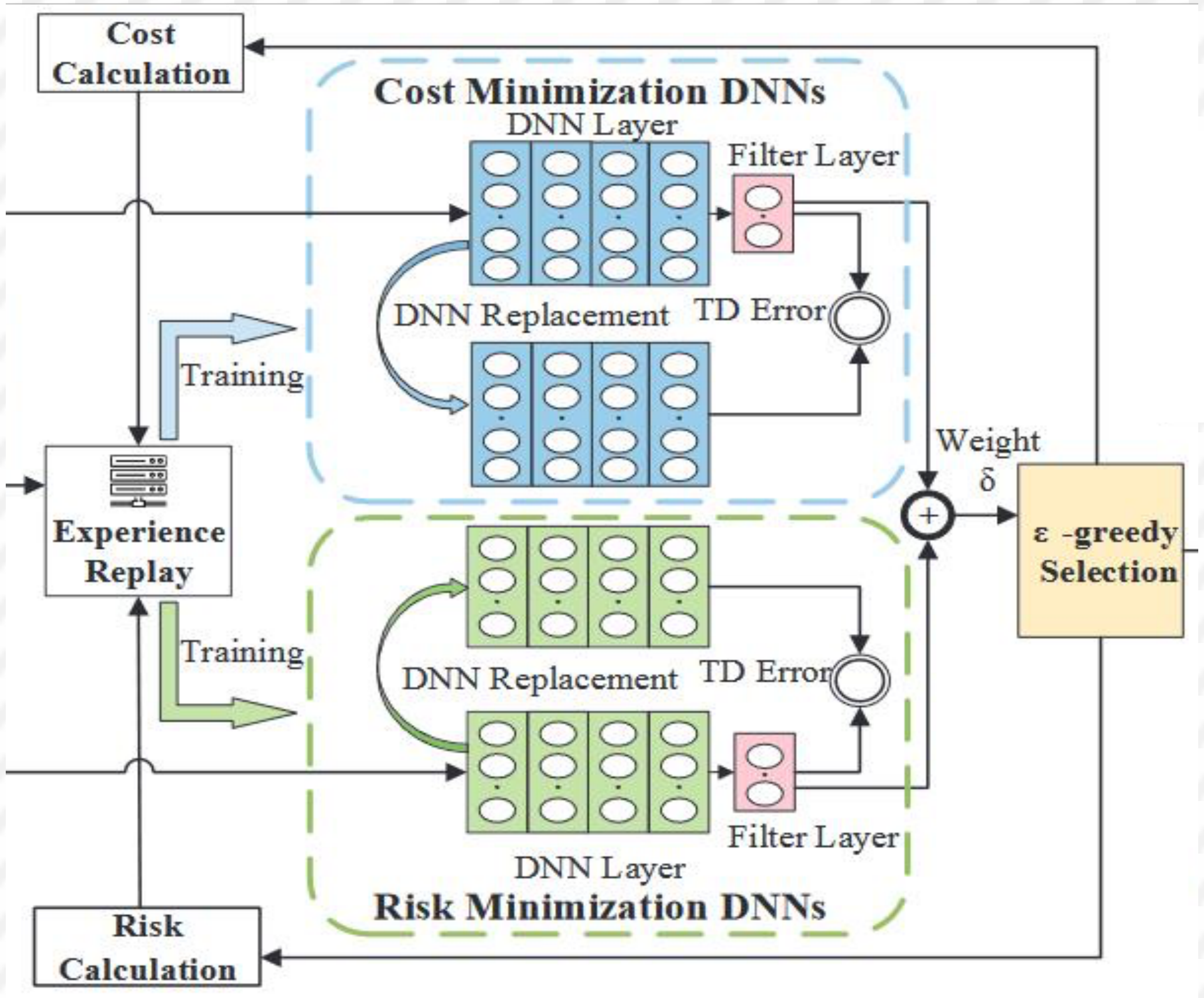
	Action a_1	Action a_2	Action a_3	Action a_4
State s_1				
State s_2				
State s_3				
⋮				

Q-Learning(DQN Version)：

- Observe a transition (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \mathbf{w})$.
- TD error: $\delta_t = Q(s_t, a_t; \mathbf{w}) - y_t$.
- Update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, Q(s_t, a_t; \mathbf{w})}{\partial\, \mathbf{w}}$.

- **Goal:** Learn the optimal action-value function $Q^\star$.
- **Tabular version** (directly learn $Q^\star$).
  - There are finite states and actions.
  - Draw a table, and update the table by Q-learning.
- **DQN version** (function approximation).
  - Approximate $Q^\star$ by the DQN, $Q(s, a; \mathbf{w})$.
  - Update the parameter, $\mathbf{w}$, by Q-learning.

Deep Risk-Sensitive RL Algorithm



Q-Learning(DQN Version)：

- Observe a transition (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \mathbf{w})$.
- TD error: $\delta_t = Q(s_t, a_t; \mathbf{w}) - y_t$.
- Update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, Q(s_t, a_t; \mathbf{w})}{\partial\, \mathbf{w}}$.

DNN-based Implementation

- Choose the action based on the sum of two Q-value functions
- DNN Replacement
- Filter Layer Design:exclude the outputs of unavailable actions
- Experience Replay
- ϵ -Greedy Selection

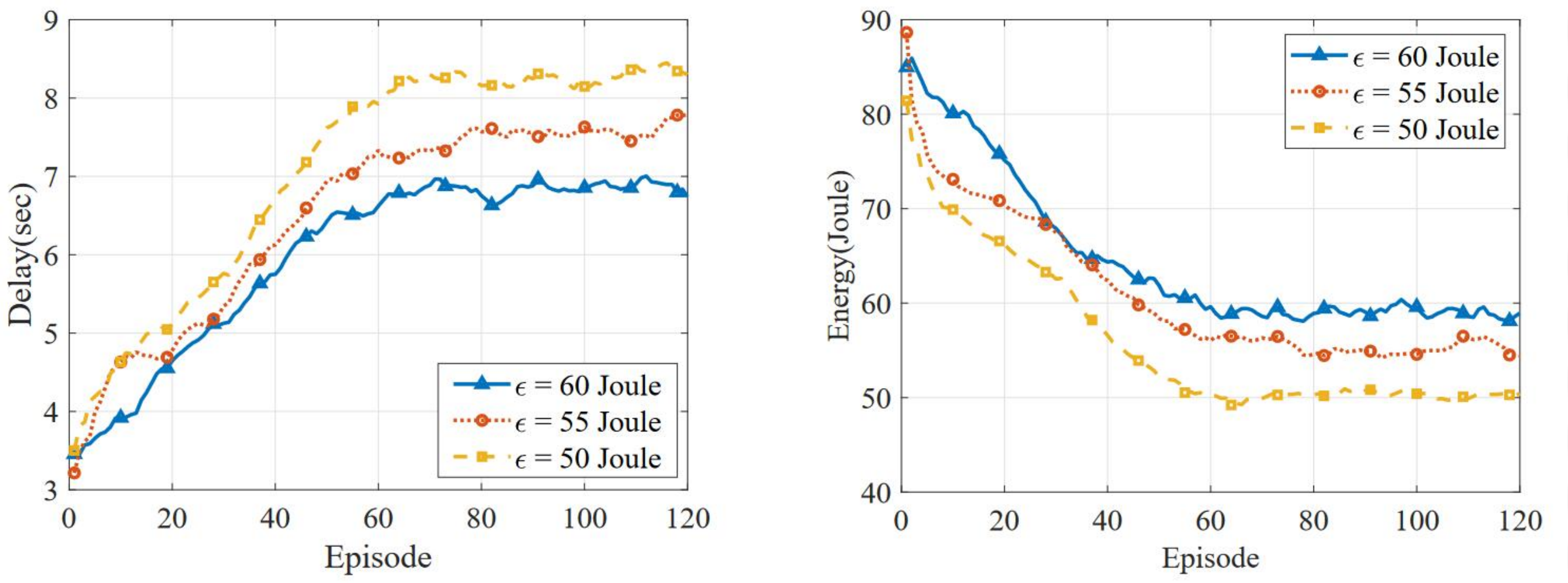
```
p = random()

if p < ε:
   pull random action
else:
   pull current-best action
```

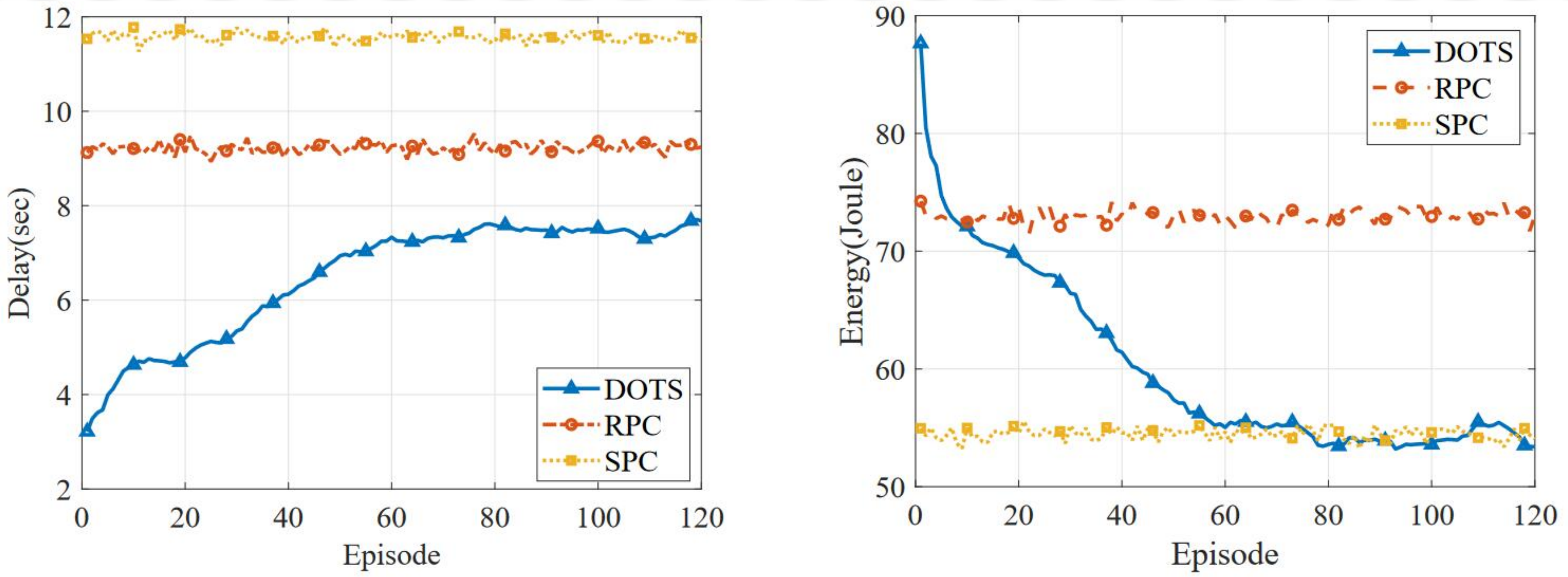
05

Performance Evalutaion & Conclusion

Performance Evalutaion

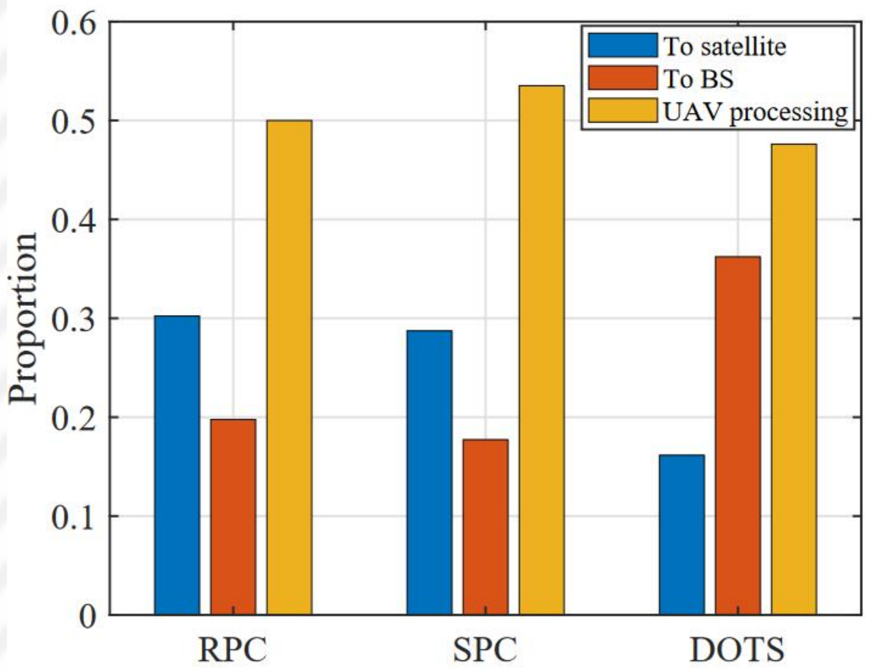


(a) (b)



(a) (b)

- **Random probabilistic configuration (RPC) 所有可用的动作都以相同的概率被选择。**
- **Sampling-based probabilistic configuration (SPC)每个state下可用action的概率是固定的**



中南大學
CENTRAL SOUTH UNIVERSITY

THANKS FOR ALL