科研题目

副标题

报告人

控制与计算机工程学院

2023 年 10 月 12 日

目录

1. 这是第一部分
2. 这是第二部分
   - 这是第二部分的第一小节
   - 这是第二部分的第二小节
   - 这是第二部分的第三小节
3. 这是第三部分
   - 使用 tikz 绘制图形
   - 使用 tikz 绘制子图
4. 这是第四部分
5. 这是第五部分

1 这是第一部分

2 这是第二部分

3 这是第三部分

4 这是第四部分

5 这是第五部分

图文分列

这是有编号的列表：

1. 使用 columns 环境
2. 使用 column 命令调整比例
3. 这里是一个插入 svg 矢量图的例子

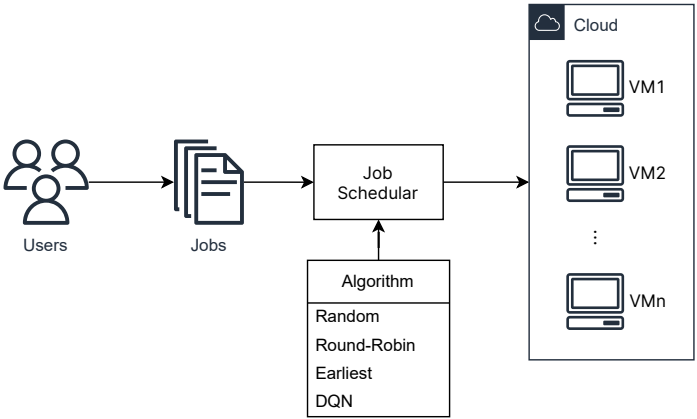


图: 这里写图片的名字

1 这是第一部分

2 这是第二部分

3 这是第三部分

4 这是第四部分

5 这是第五部分

1 这是第一部分

2 这是第二部分
这是第二部分的第一小节
这是第二部分的第二小节
这是第二部分的第三小节

3 这是第三部分

4 这是第四部分

5 这是第五部分

frame 的主标题[3]

frame 的副标题

这是一个嵌套列表：

- 111
- 222
- 333[1]
  - 3.111
  - 3.222[2]

[1]这里有一条脚注
[2]这里还有一条脚注
[3]标题脚注

1 这是第一部分

2 这是第二部分
- 这是第二部分的第一小节
- 这是第二部分的第二小节
- 这是第二部分的第三小节

3 这是第三部分

4 这是第四部分

5 这是第五部分

写点数学公式

当观察次数 $n \to \infty$ 时，表示在单位时间内持续观察用户提交任务的过程：

$$
\begin{aligned}
P\{X=k\} &= \lim_{n\to\infty} C_n^k p^k (1-p)^{n-k} \\
&= \lim_{n\to\infty} \frac{n(n-1)\dots(n-(k-1))}{k!} \left(\frac{\lambda}{n}\right)^k \left(1-\frac{\lambda}{n}\right)^{n-k} \\
&= \frac{\lambda^k}{k!} \lim_{n\to\infty} \frac{n(n-1)\dots(n-(k-1))}{n^k} \left(1-\frac{\lambda}{n}\right)^{-k} \left(1-\frac{\lambda}{n}\right)^{n} \\
&= \frac{\lambda^k}{k!} e^{-\lambda}
\end{aligned}
$$

此时单位时间内实际提交的任务数量 X 服从泊松分布 $X \sim P(\lambda)$ 。

1 这是第一部分

2 这是第二部分
- 这是第二部分的第一小节
- 这是第二部分的第二小节
- 这是第二部分的第三小节

3 这是第三部分

4 这是第四部分

5 这是第五部分

block 示例

记 $[0,t]$ 时间段内用户提交的任务数量为 $N(t)$，$(s,t]$ 时间段内用户提交的任务数量为 $N(s,t]=N(t)-N(s)$。

泊松过程

- $N(0)=0$：初始时刻没有用户提交任务
- 独立增量性：在互不相交的时间段内，用户提交任务的数量相互独立
- 平稳增量性：在长度相等的时间段 t 内，任务提交数量服从相同的概率分布 $P(\lambda t)$

因此 $\forall s, N(s,s+t]\sim P(\lambda t)$：

$$P\{N(s,s+t]=k\}=\frac{(\lambda t)^k}{k!}e^{-\lambda t}$$

1 这是第一部分

2 这是第二部分

3 这是第三部分

4 这是第四部分

5 这是第五部分

1 这是第一部分

2 这是第二部分

3 这是第三部分
使用 tikz 绘制图形
使用 tikz 绘制子图

4 这是第四部分

5 这是第五部分

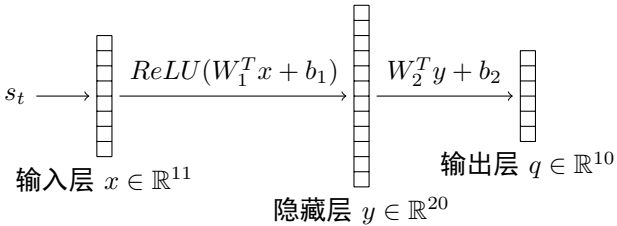
用 tikz 画了个图

这部分内容放在了另外一个 tex 文件中

代码默认存在 10 台虚拟机，输入状态 s_t 为当前提交任务的类型和 10 台虚拟机的 T_{idle} ：

$$s_t = \left[Type, T_{idle}^{(1)}, T_{idle}^{(2)}, \ldots, T_{idle}^{(10)}\right]^T$$

DQN输出在当前状态 s_t 下，对分配到每台虚拟机的总共 10 个动作的评分。



其中两层全连接层的参数：$W_1 \in \mathbb{R}^{11\times20}$, $b_1 \in \mathbb{R}^{20}$, $W_2 \in \mathbb{R}^{20\times10}$, $b_2 \in \mathbb{R}^{10}$ 。

1 这是第一部分

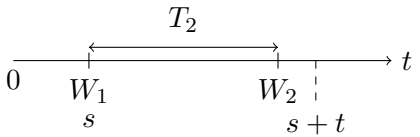
2 这是第二部分

3 这是第三部分
使用 tikz 绘制图形
使用 tikz 绘制子图

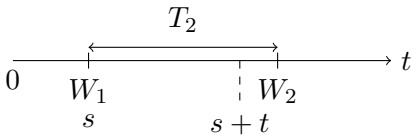
4 这是第四部分

5 这是第五部分

用 tikz 花了两个子图



(a) $T_2 \leqslant t$



(b) $T_2 > t$

1 这是第一部分

2 这是第二部分

3 这是第三部分

4 这是第四部分

5 这是第五部分

试试代码高亮

记得加 fragile 参数

这个字是正常大小

```
# scriptsize 命令变成脚本大小
# 生成时间间隔
intervalT = stats.expon.rvs(scale=1 / lamda, size=self.jobNum)
# 对时间间隔累加得到提交时间
self.arrival_Times = np.around(intervalT.cumsum(), decimals=3)
```

又恢复了正常大小

```
# 又变成了脚本大小
self.jobsMI = np.random.normal(self.jobMI, self.jobMI_std, self.jobNum)
self.jobsMI = self.jobsMI.astype(int)
```

1 这是第一部分

2 这是第二部分

3 这是第三部分

4 这是第四部分

5 这是第五部分

最后试试伪代码

shrink 参数调整 frame 的缩小系数

设定环境参数、DRL超参数，随机初始化DQN参数，赋予target网络相同的参数;
foreach *Episode* **do**
    重置环境;
    foreach *Step* **do**
        对于状态 s_t 根据 ϵ-greedy 策略得到动作 a_t;
        执行动作 a_t 后环境变为 s_{t+1} 并得到奖励 r_t;
        将轨迹 (s_t, a_t, r_t, s_{t+1}) 存入replay memory;
        if *Step* $>$ 开始学习步数 **then**
            从replay memory中随机抽取 30 个样本作为minibatch;
            foreach *sample in minibatch* **do**
                将 s_t 传入Q-network得到 a_t 对应的 Q_{value};
                将 s_{t+1} 传入target-network得到输出的最大值 Q_{target};
                根据损失函数 $Loss = (Q_{value} - (r_t + \gamma \cdot Q_{target}))^2$ 使用梯度下降法更新Q-network;
            end
            if *Step % 50 = 0* **then**
                使用Q-network的参数更新target-network;
            end
            减小 ϵ;
        end
    end
end

Thanks for listening!