

# **Enabling Scalable Routing in Software-Defined Networks With Deep Reinforcement Learning on Critical Nodes**

**TON 21**

**DML GROUP MEETING  
3.18**

# OUTLINE

- Introduction
- Design
- Evaluation

# Introduction

- 传统路由方案通常使用固定的路由策略模型，因此不擅长处理复杂的动态流量
- 最新研究：DRL 与 SDN 相结合，通过自动流量分析和策略生成提高网络性能
- 存在的问题：通常依赖于所有节点信息来为网络做出路由决策
  - 1) 在大型网络中难以收敛 (Curse of Dimensionality)
  - 2) 容易受到拓扑变化的影响 (Low robustness)
- 改进方案：提出 ScaleDeep，一种可扩展的基于 DRL 的 SDN 路由方案
- ScaleDeep关键思想：
  - 1) 基于控制理论，从网络中选择一组关键节点作为driver节点，可以模拟整个网络的运行
  - 2) 通过观察driver节点上的流量变化，DRL动态调整链路权重

# Opportunity and Challenges

- 互联网是一个复杂的无标度网络，控制每个节点以达到优化目标是不切实际的
- 现有研究：Pinning control
- 选择性地控制网络中的一些关键节点 (**driver nodes**)，以实现类似于协调整个网络中所有节点
- 采用pinning control的两个需要解决的问题：
  - 1) 如何为给定拓扑选择driver节点？
  - 2) 如何控制driver节点？Pinning control通常使用控制信号来调整driver节点。现有的Pinning control方案并未清楚说明driver节点上的输入控制信号会如何影响性能

# Design

- 分为offline和online两部分
- offline:
  - 1) 根据pinning control理论从网络中选择一个路由节点子集
  - 2) DRL 代理对driver节点施加直接控制, follower节点在driver节点的影响下改变路由行为
  - 3) DRL代理作为SDN控制器上方的应用运行以通过学习路由策略来更新其神经网络参数并生成路由信号来将路由策略更新到网络中来与网络交互
- online:
  - 1) 控制器使用来自 SDN 的现有收集技术从网络收集driver节点的状态 (即驱动节点端口的吞吐量) 和网络性能, 并将收集的信息发送到 DRL 代理。
  - 2) DRL 代理更新网络链接权重。DRL 将第一步收集的信息作为其输入, 并为网络生成新的链路权重作为动作, 然后将新的链路权重信息发送到 SDN 控制器。
  - 3) 控制器使用加权最短路径算法计算转发路径, 并通过将流条目安装到相关路由节点来部署这些路径

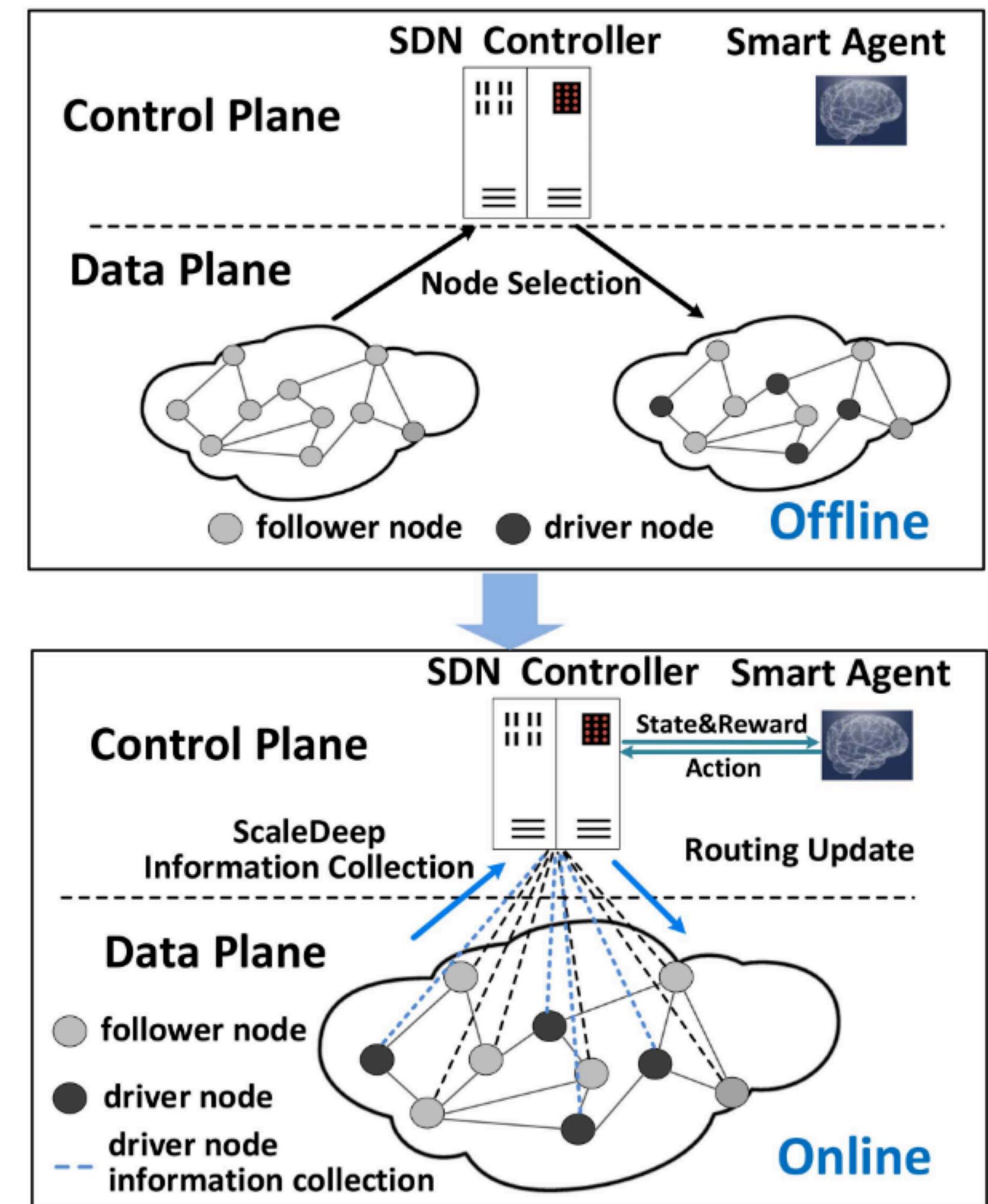


Fig. 1. Architecture of ScaleDeep.

# Design

## Pinning Control Theory and Principles for Node Selection

- Pinning control theory 目标：选择网络中的 driver 节点，该节点是模拟网络运行的关键节点，其余节点是 follower 节点
- 如何选择驱动程序节点？
- 现有工作：刘等人 [23] 提出了网络中 driver 节点数量和位置选择的两个分析结论
  - 1) 在具有相同的入度指数和出度指数的无标度网络中，网络中驱动器节点  $n_D$  的比例可近似计算如下，其中  $k$  表示网络中节点的平均度数。
$$n_D \approx \exp\left[-\frac{1}{2}\left(1 - \frac{1}{\gamma - 1}\right)\langle k \rangle\right]$$
  - 2) 在全连接网络中，驱动节点倾向于避开高度节点。同时，只有一个度的节点不能成为驱动节点，因为它们对其他节点的影响有限。
- ScaleDeep 采用这两个结论作为原则来启发式地选择网络中的驱动节点

# Design

## Driver Node Selection Algorithm

- 启发式算法：基本思想是根据度数分配不同概率的节点，然后根据概率选择驱动节点
- 算法步骤：
  - 1) 节点按度数升序排序
  - 2) 选择follower节点：度数为 1 的节点被标记为follower节点，因为控制它只能调整流经它的流的路径。如果follower节点的邻居节点的度数仅为 2，则这些节点也被标记为follower节点
  - 3) 交替选择driver节点和follower节点
    - 第 14 行将 CV 中的所有元素重置为零以用于新更新的拓扑。
    - 第 15-19 行类似于第 3-4 行选择跟随者节点。
    - 第 20-23 行为不同程度的节点分配不同的选择概率。
    - 第24-26行根据其概率选择一个驱动节点并将其添加到驱动节点集合D中。
    - 第27行将新选择的驱动节点的邻居节点选择为跟随节点并将它们添加到跟随节点集合中。
    - 第 28 行通过删除新选择的驱动节点和跟随节点及其链接来更新拓扑。

---

### Algorithm 2 Driver Node Selection Algorithm

---

```
Input: Network topology  $G = (V, E)$ ;  
Output:  $D$ : the set of driver nodes,  $F$ : the set of follower nodes.  
1: Sort nodes in  $V$  following the ascending order of their degrees; 节点按度数升序排序  
2: for  $v \in V$  do 选择follower节点  
3:   if  $d_v == 1$  then  
4:      $F = F \cup v$ ;  
5:   for  $v_{Nb} \in Nb(v)$  do  
6:     if ( $v_{Nb} == 2$ ) then  
7:        $F = F \cup v_{Nb}$ ;  
8:     end if  
9:   end for  
10: end if  
11: end for  
12: Remove the nodes of  $F$  from  $V$  and update  $G$ ; 交替选择driver节点和follower节点  
13: while  $V \neq \emptyset$  do  
14:   Reset all elements in  $C_V$  to 0;  
15:   for  $v \in V$  do  
16:     if  $d_v == 1$  then  
17:        $F = F \cup v$ ;  
18:     end if  
19:   end for  
20:   for  $v \in V$  do  
21:      $C_V(d_v) = C_V(d_v) + 1$ ;  
22:     Set probability value  $P(d_v) = \frac{C_V(d_v)}{\sum_{d_v} C_V(d_v)}$ ;  
23:   end for  
24:   Randomly pick degree value  $j$  with  $P(j)$ ;  
25:   Randomly pick node  $v$  such that  $d_v == j$ ;  
26:    $D = D \cup v$ ;  
27:    $F = F \cup Nb(v)$ ;  
28:    $V = V - F - D$ , and remove links that connect to  $v$  and  $Nb(v)$  from  $E$ ;  
29: end while
```

# Design

## DRL Background and Mathematical Model

- DRL 算法：DDPG，使用了两种类型的神经网络：**GRU 和前馈神经网络**。网络流量通常具有**与时间相关的信息**，GRU 善于从输入数据中提取与时间相关的信息。
- State：状态是网络状态信息，由大小为  $t \times n$  的吞吐量矩阵表示，其中  $t$  表示时间步长， $d$  表示流类型的数量， $n$  表示流量强度的总数从驱动节点收集
- reward：使用网络中所有流的平均流完成时间作为奖励  $r$
- Action：添加正态分布的噪声值 ( $N$ )，并将动作的值限制在一个固定的范围内，以提高 DRL 在训练过程中的稳定性

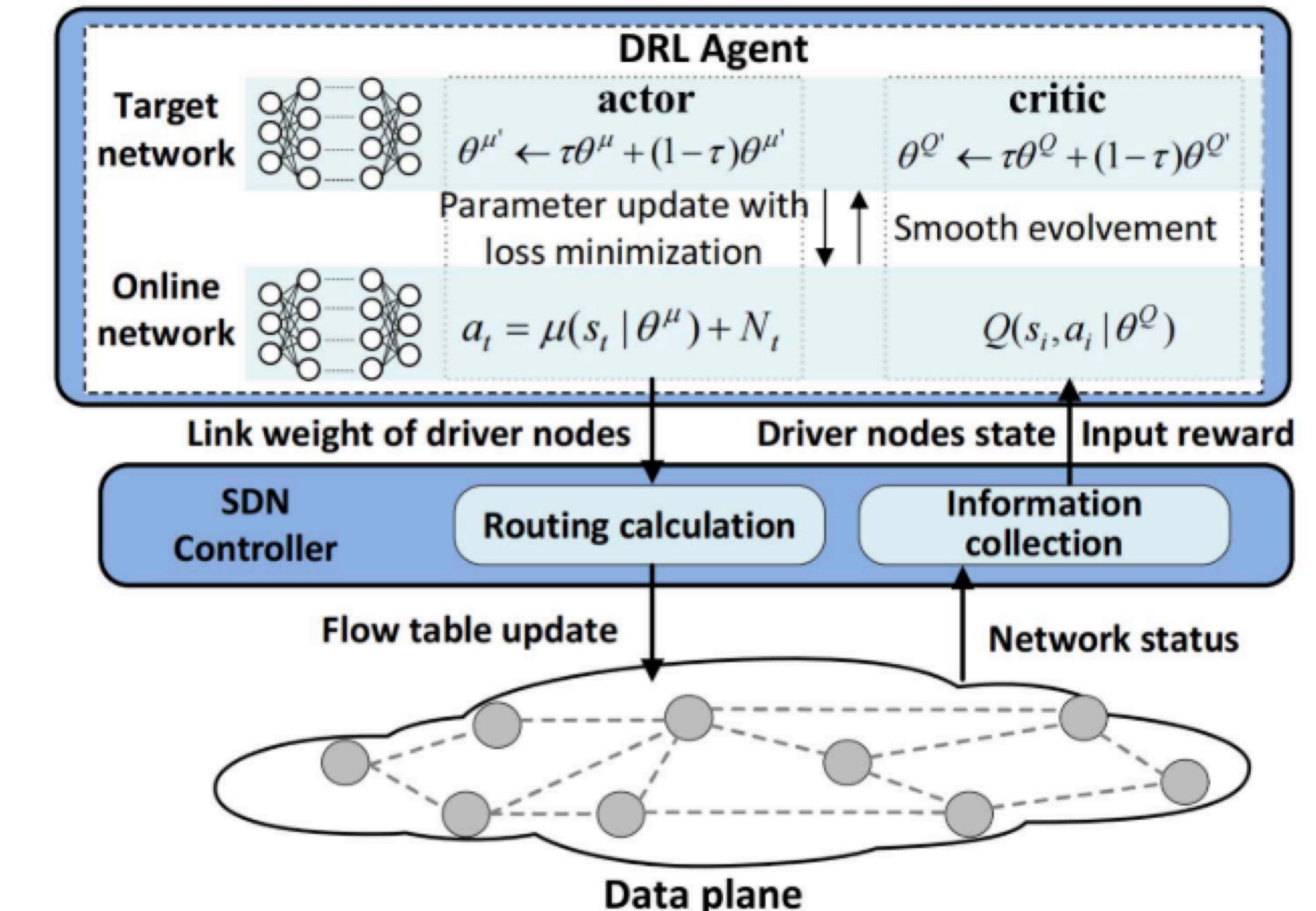


Fig. 2. DRL control logic of ScaleDeep.

# Evaluation

- 目的：比较LSTM、GRU和FFN组合的效果
- 效果：LSTM 或者GRU与前馈神经网络的串联明显优于纯前馈神经网络
- 原因：LSTM和GRU具有更好的分析网络流量时间相关性的能力。LSTM需要更多的训练时间，因为它有更多的计算参数，但GRU和LSTM有相似的路由性能。因此，最终选择GRU而不是LSTM。

TABLE III  
AVERAGE FCT UNDER DIFFERENT NEURAL NETWORK COMPOSITION

3-layer FFN	LSTM + 2-layer FFN	GRU + 1-layer FFN	GRU + 1-layer FFN	GRU + 3-layer FFN
9.35ms	7.34ms	7.85ms	7.33ms	7.56ms

# Evaluation

- 目的：比较driver节点比例的影响
- 实验：分别选择了 9、13 和 15 个驱动节点，三种方案的平均流完成时间为 59ms, 50 毫秒和 52 毫秒
- 结论：更多的驱动节点不一定会带来更好的 ScaleDeep 性能。但是到目前为止，还没有一种理论方法来为给定的拓扑选择最优的driver节点

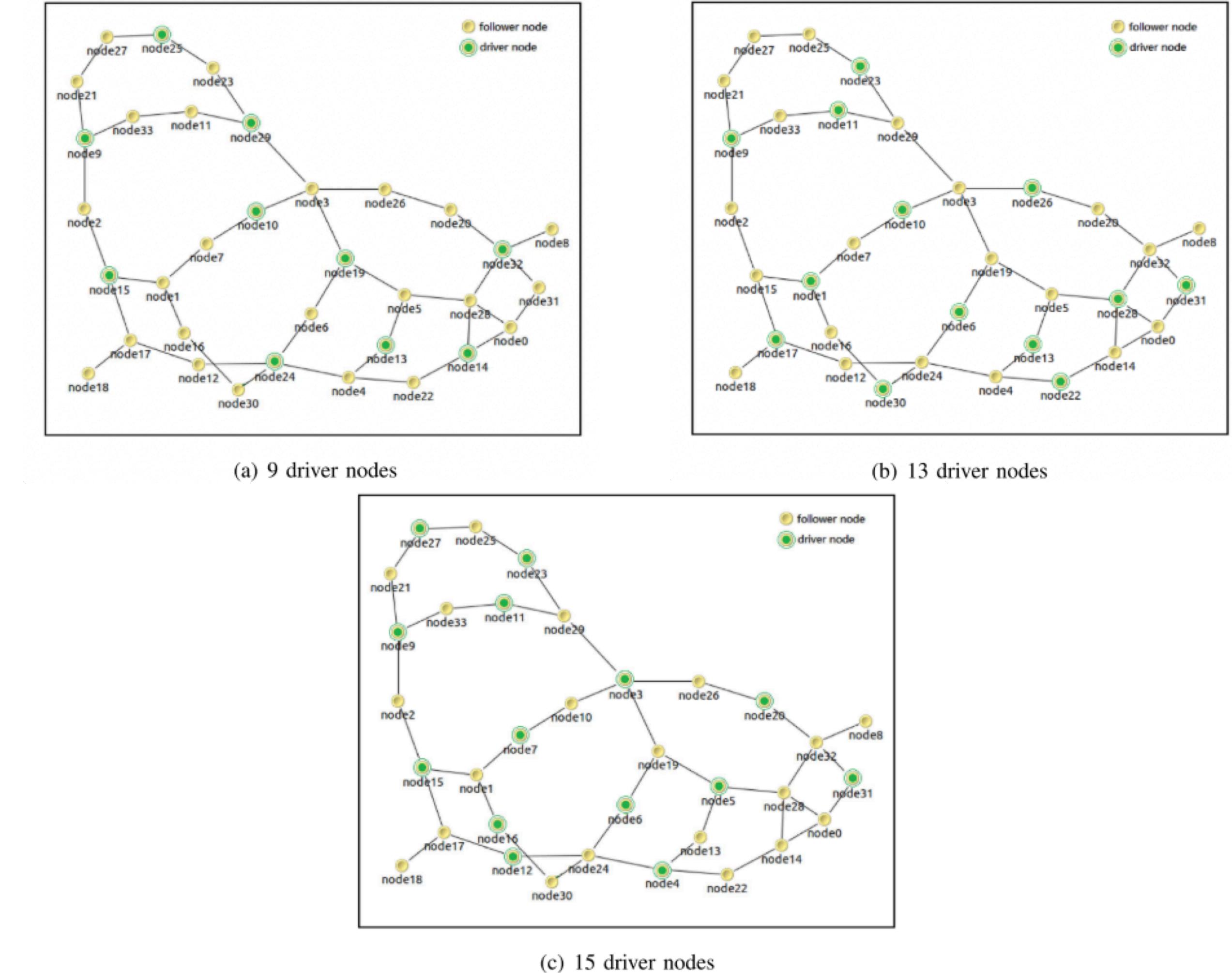
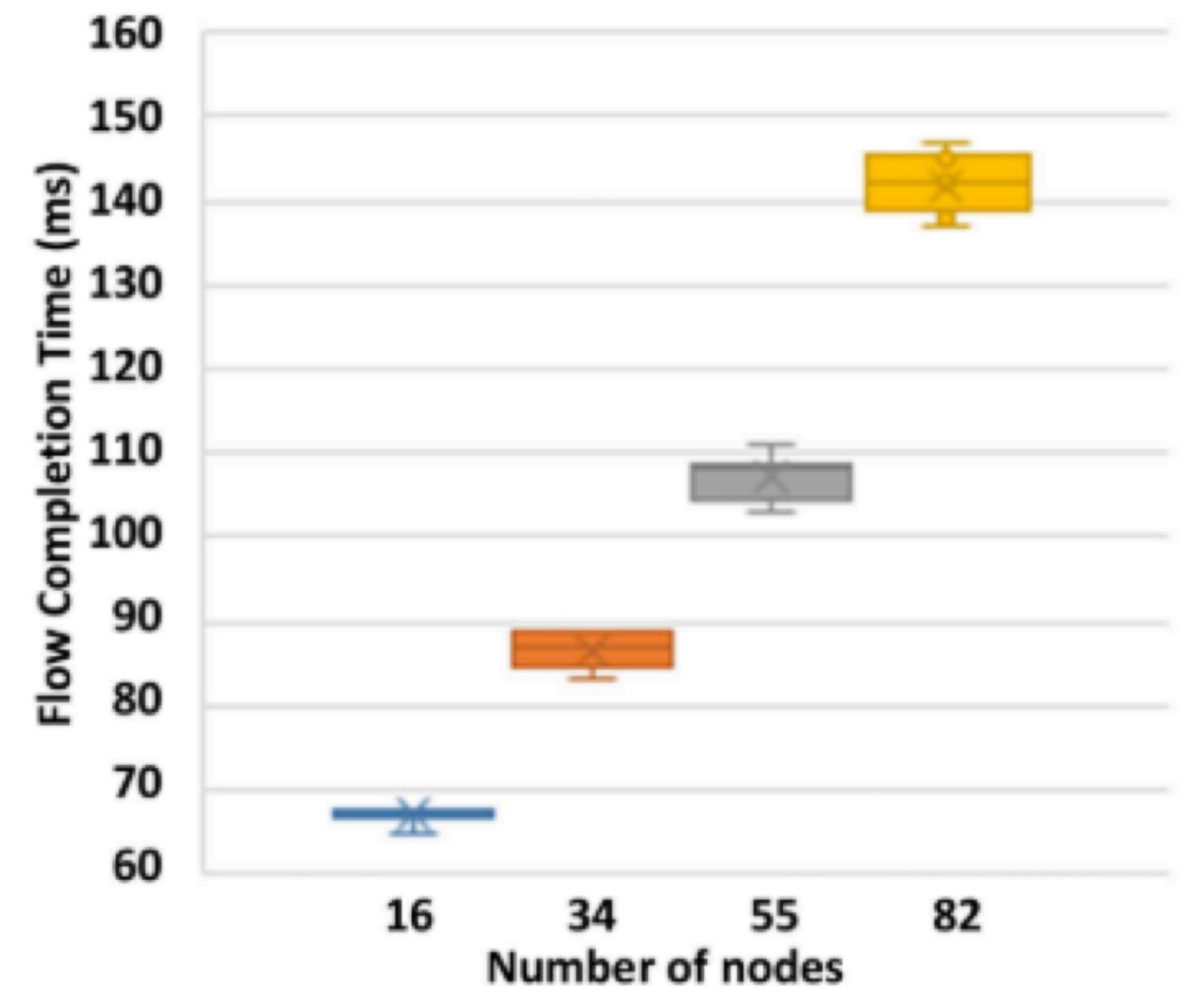


Fig. 3. Different selection schemes of driver nodes under OS3E.

# Evaluation

- 目的：观察driver节点选择算法的性能稳定性
- 实验：给定相同的输入拓扑，运行该算法可以生成不同的驱动节点集。对于 16、34、55 和 82 个节点的拓扑，运行选择算法10 次以生成不同的驱动节点集，并使用它们来测试 ScaleDeep 的性能稳定性
- 结果：在图中，各拓扑与中值的偏差在5%以内。因此，ScaleDeep 的性能表现出稳定性



# Evaluation

- 目的：泛化能力分析
- 实验：在模拟过程中改变了流量的 RP 比率  
( $RP = RF / PF$ , 随机流/周期流)
- 结果：当流量从  $RP = 0.1$  变为  $RP = 0.2$  (在时间  $t_1$ ) 和从  $RP = 0.2$  变为  $RP = 0.05$  (在时间  $t_2$ ) 时，ScaleDeep 可以快速适应网络流量变化，保持相对稳定的性能

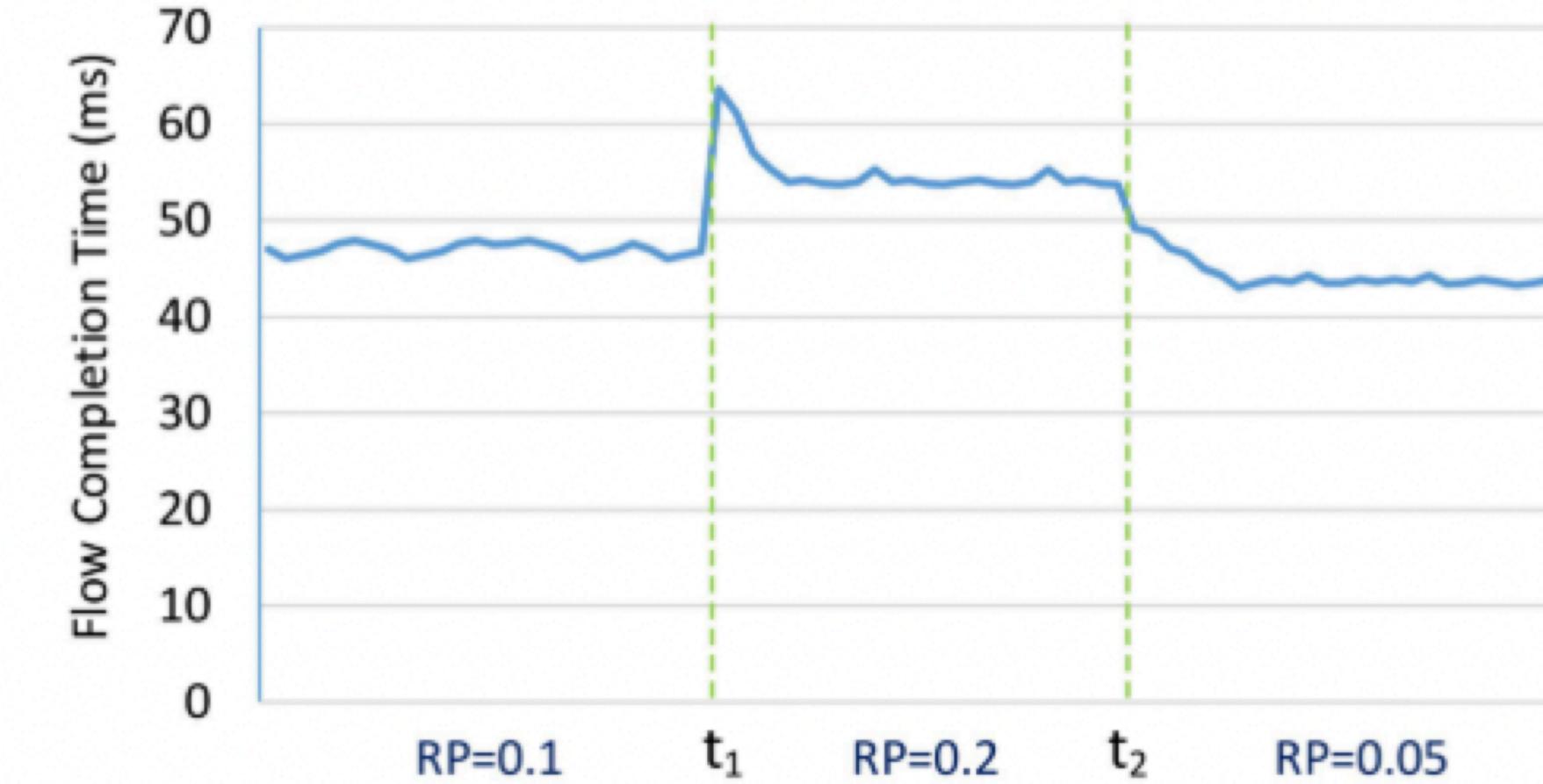


Fig. 5. The performance of ScaleDeep under a changing traffic pattern.

# Evaluation

- 目的：比较ScaleDeep的性能
- 展示了上传消息的数量（即用于状态拉取和奖励计算的消息数量）和下载消息的数量（即用于更新路由策略的消息数量）。上传和下载消息显示SDN控制器的开销，与上传和下载频率成正比。
- 结果：在所有方案中，DRL-TE 的开销最大，因为 DRL-TE 必须为每个流生成控制动作。ScaleDeep 请求控制器上的开销最少，因为它仅控制网络中的驱动程序节点以进行状态拉取和路由策略更新。
- 在这个测试场景中，DRLTE、DDPG-R和ScaleDeep的FCT分别为8.6ms、9.9ms和7.1ms。与 DRL-TE 相比，ScaleDeep 将流程完成时间减少了约 25%。

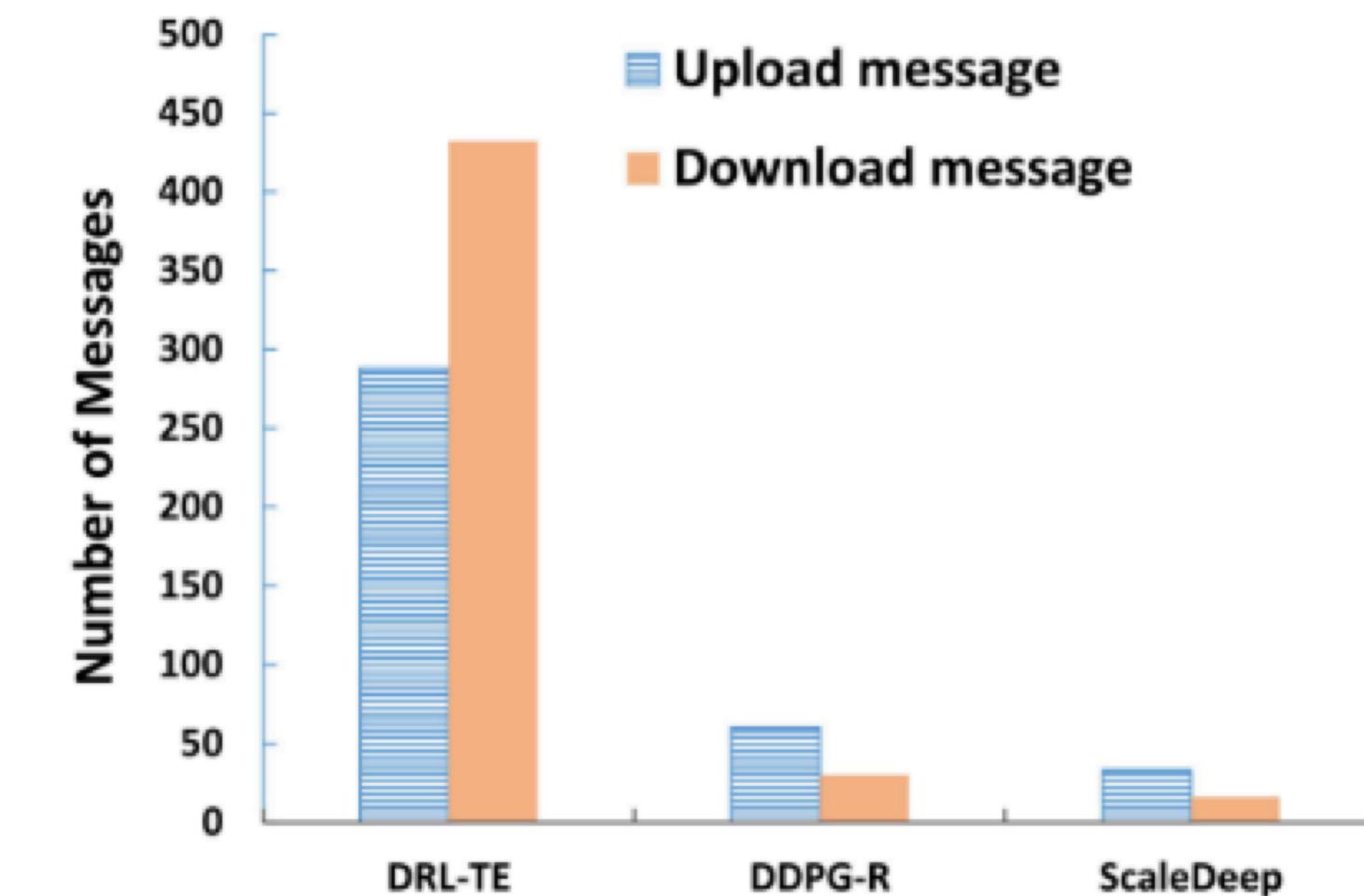
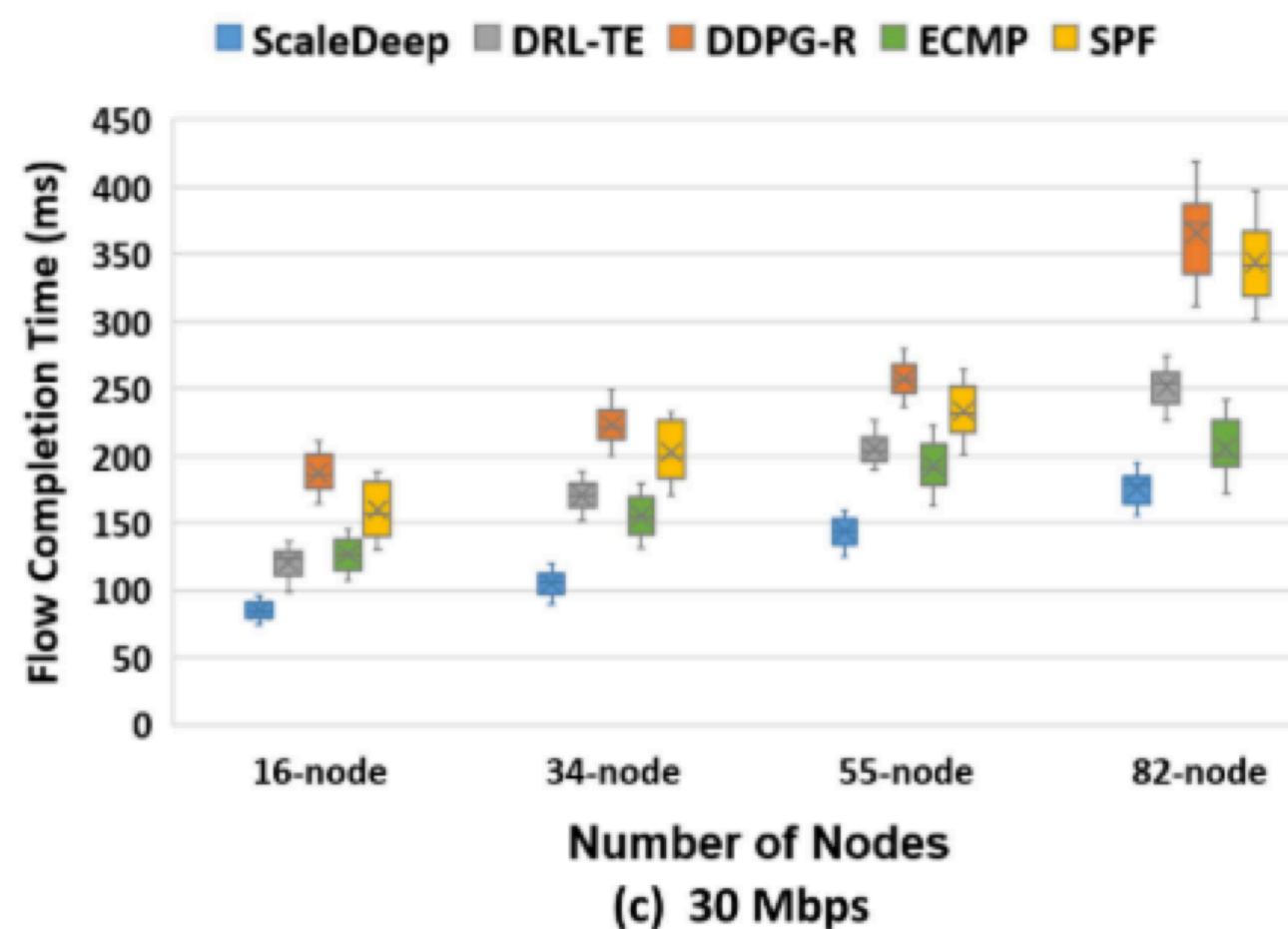
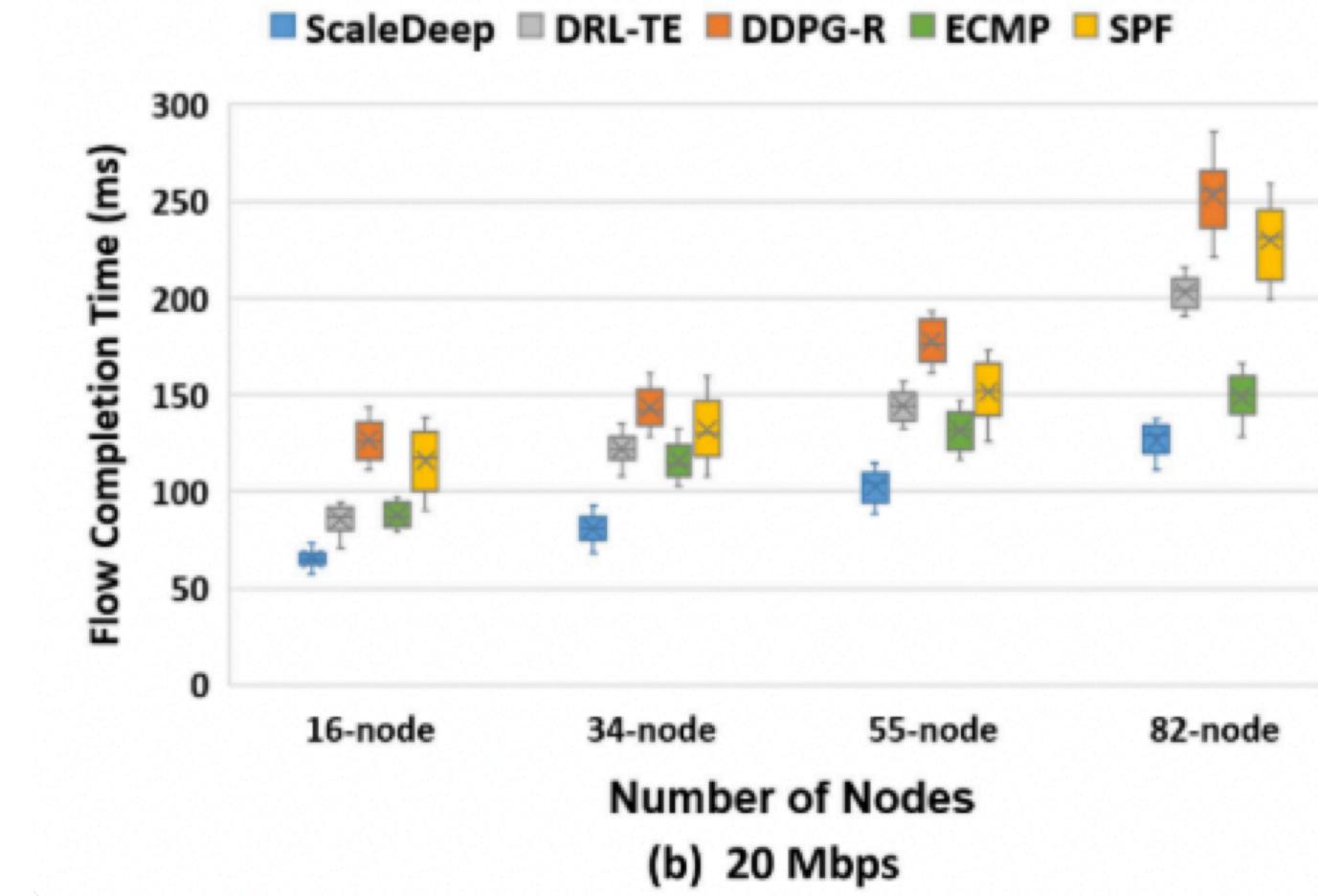
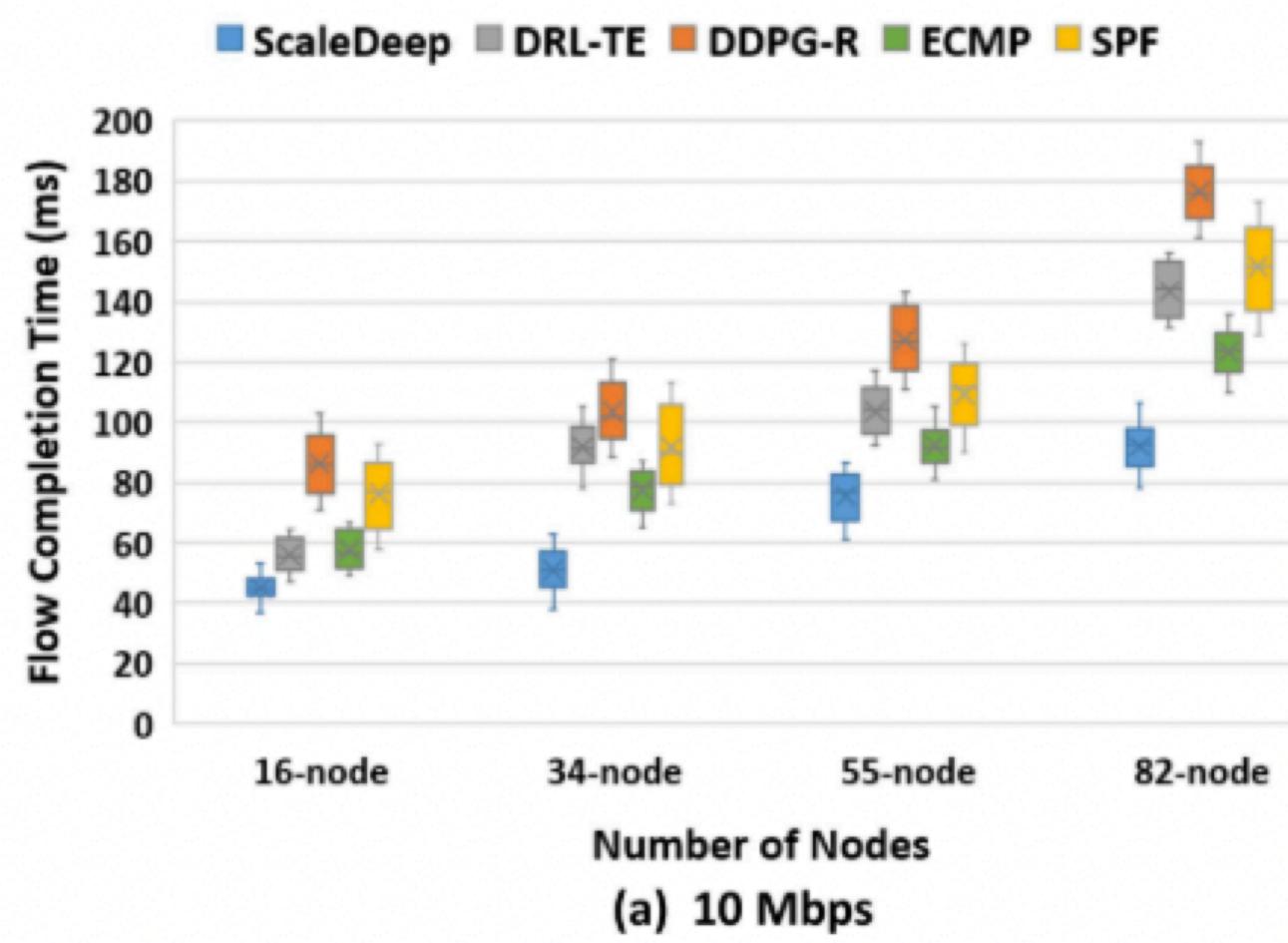


Fig. 6. Communication overhead under Abilene network. The upload messages are used for state pulling and reward calculation, the download messages are used for updating the link weights.

# Evaluation

- 目的：比较四种方案在不同总流量和网络规模下的网络性能



# Evaluation

- 显示了奖励变化过程。ScaleDeep 的训练奖励长期保持增长
- 原因：当网络规模较大时，由于使用 pinning control theory所需的控制信息较少，ScaleDeep 不太可能遇到维度灾难。相反，DDPG 和 DRL-TE 遭受维数灾难，无法从训练中获得足够的有用信号来进行参数演化。

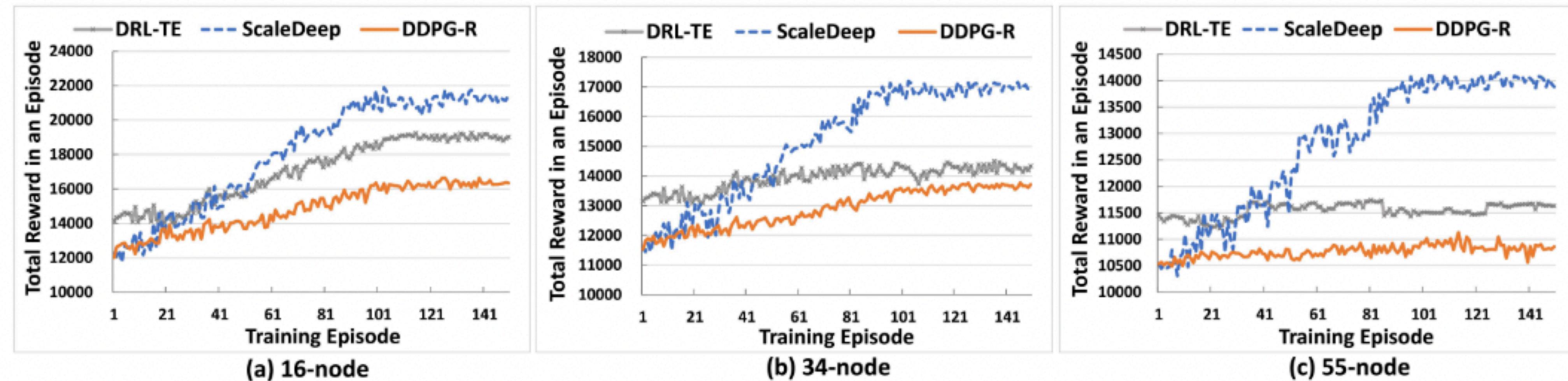


Fig. 8. Training process under the different topologies. Each training episode contains 1000 training steps.

# Evaluation

- 显示了三种基于 DRL 的路由方案的鲁棒性结果
- 在图中，当节点数量增加或减少时，DRL-TE 和 DDPG 的性能会显著下降。相比之下，ScaleDeep 在节点的不同变体中保持了相对稳定的性能
- ScaleDeep 的鲁棒性主要适用于追随者节点。如果一个follower节点出现故障，ScaleDeep的输入格式不会受到影响，因此ScaleDeep仍然可以正常工作。如果驱动节点出现故障，ScaleDeep 也需要像其他基于 DRL 的路由方案一样重新训练。

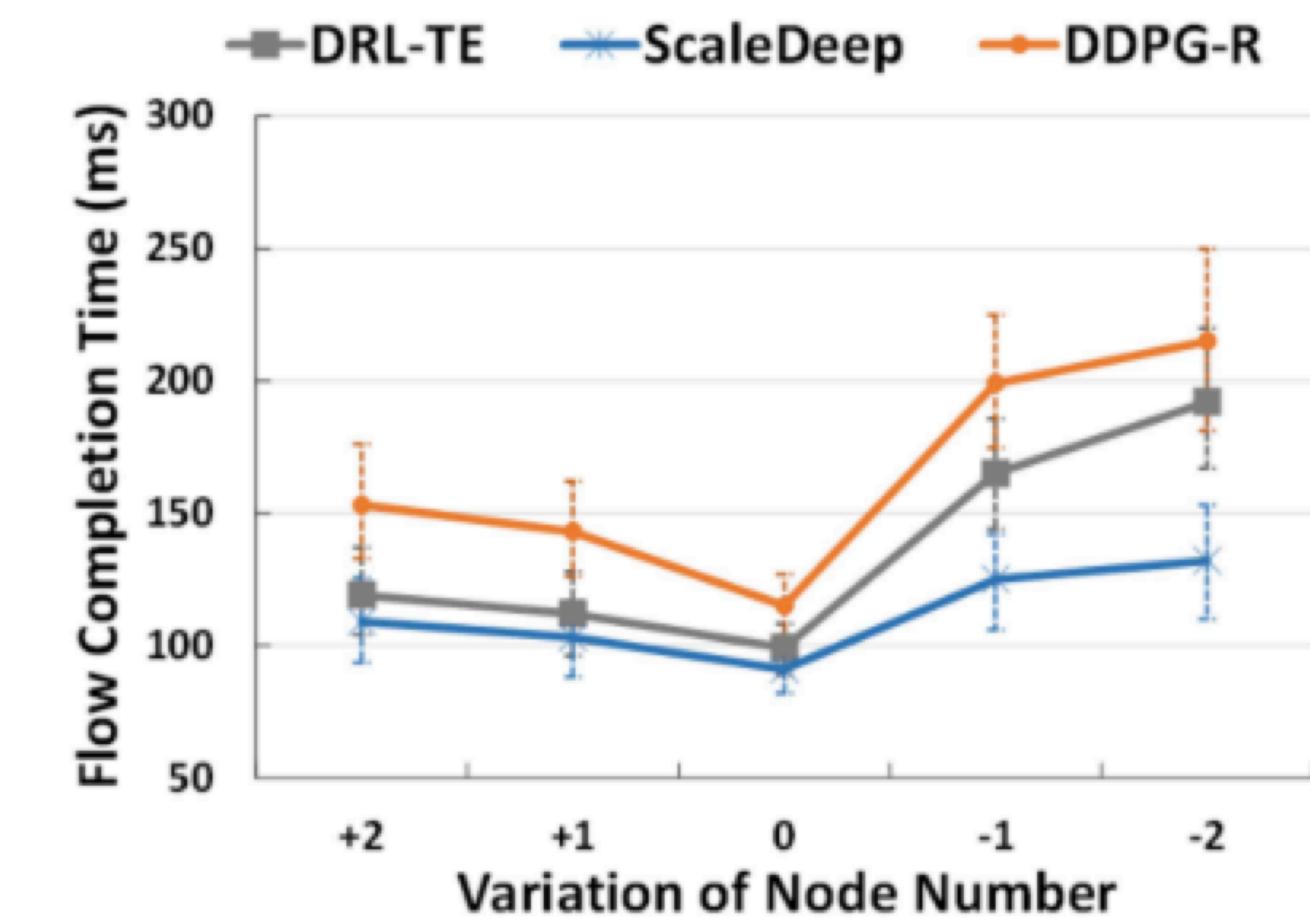


Fig. 9. Robustness under the 55-node topology. (+1 means adding a new node and -1 means removing a node from the network.)

# Evaluation

- 不同RP流量比的影响，不同RP比下不同方案的性能。

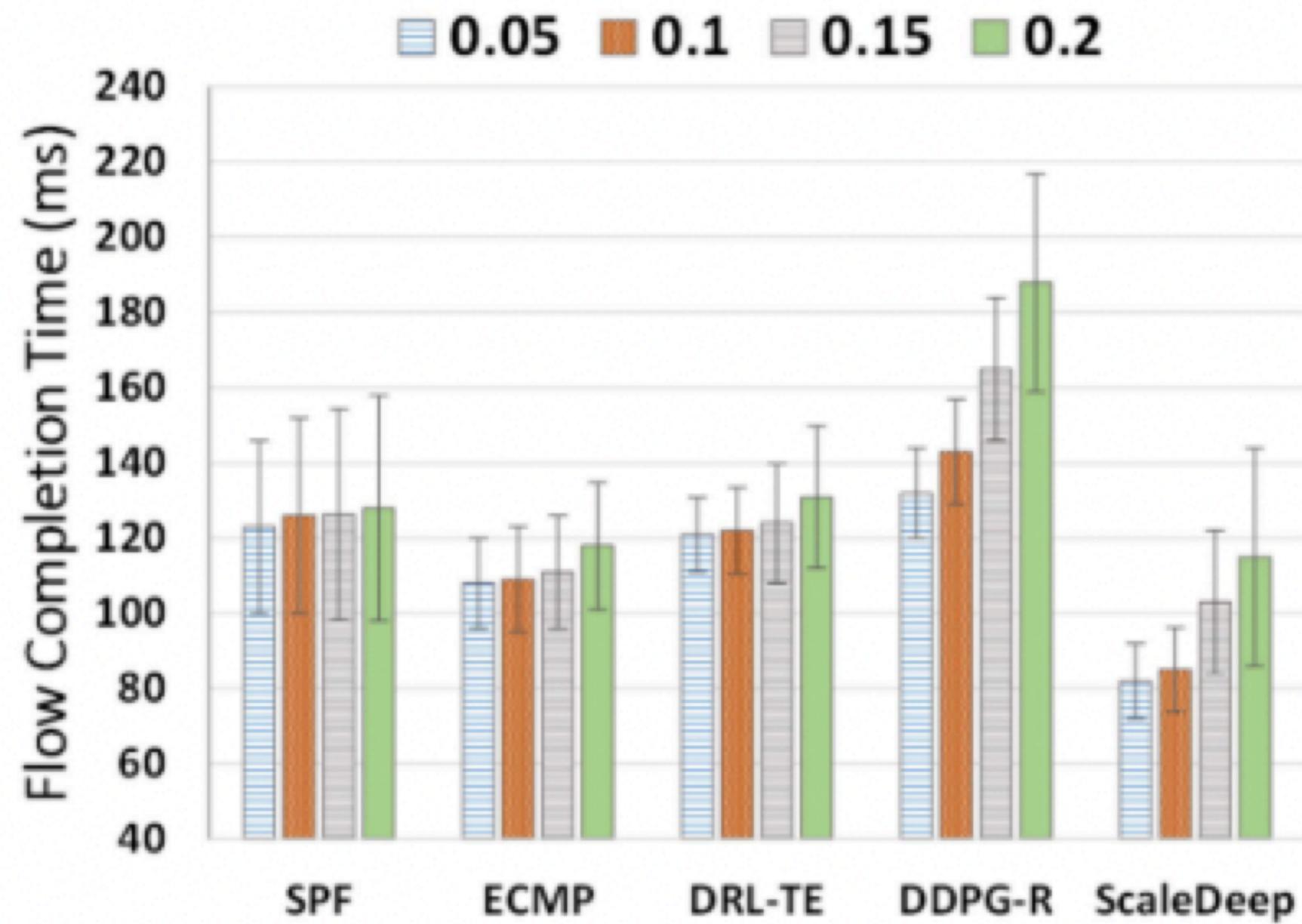


Fig. 10. Flow completion time of ScaleDeep under different RP ratios.

## VII. RELATED WORK

# Conclusion

- 缺点：
  - 1) 采用的是启发式算法可能得不到最优解，是通过实验证明稳定性

**THANKS!**