面向虚拟数据空间的智能 TCP 拥塞控制算法

王龙翔[1]，董凯[2]，李小轩[1]，董小社[1]，张兴军[1]，朱正东[1]，王宇菲[1]，张利平[1]
（1. 西安交通大学计算机科学与技术学院，710049，西安；2. 西安美术学院信息中心，710065，西安）

摘要：为优化虚拟数据空间网络传输性能，提出了基于近端策略优化的智能 TCP 拥塞控制算法 TCP-PPO2。将 TCP 拥塞控制过程抽象为一个可部分观察的马尔可夫决策过程，在该过程中构建一个智能体，与网络环境进行互动。智能体通过观察网络状态特征对拥塞窗口长度进行调节，网络环境向智能体反馈奖励值，智能体尝试最大化回合内获得奖励期望值。设计了包括吞吐率、网络时延等网络特征的状态空间，使智能体能够观察到足够多的信息进行决策并且降低性能开销。通过加权算法设计奖励函数，使智能体能够平衡优化吞吐率与时延。通过近端策略优化算法更新智能体模型参数，对过大的参数更新进行截断，将参数更新限制在一定范围内，减少梯度下降过程中出现的振荡，实现训练过程的快速收敛。在 NS3 模拟器上实现了基于近端策略优化的 TCP 拥塞控制算法，并与 Cubic、HighSpeed 和 NewReno 等主流拥塞控制算法进行了对比，结果表明：TCP-PPO2 吞吐率性能可达对比算法的 2～3 倍以上；80%的采样点时延相比链路最小时延值只增加了 4%。

关键词：虚拟数据空间；近端策略优化；拥塞控制；TCP
中图分类号：TP319 **文献标志码**：A
DOI：10.7652/xjtuxb202105010 **文章编号**：0253-987X(2021)05-0083-09

OSID 码

Intelligent TCP Congestion Control Method for Virtual Data Space

WANG Longxiang[1], DONG Kai[2], LI Xiaoxuan[1], DONG Xiaoshe[1], ZHANG Xingjun[1], ZHU Zhengdong[1], WANG Yufei[1], ZHANG Liping[1]
(1. School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China;
2. Information Center, Xi'an Academy of Fine Arts, Xi'an 710065, China)

Abstract: To optimize the transmission performance of virtual data space network, an intelligent TCP congestion control algorithm based on proximal policy optimization is proposed (TCP-PPO2). The TCP congestion control process is abstracted as a Markov decision process, which can be partially observed. In this process, an agent is constructed to interact with the network environment. The agent adjusts the size of congestion window by observing the characteristics of network state. The network environment feeds back a reward value to the agent, and the agent tries to maximize the expected reward value in an episode. The state space including throughput, network delay and other network characteristics is designed, so that agents can observe enough information to make decisions and reduce performance overhead. The weighted reward function is designed to balance the throughput and delay. The parameters of the agent model are updated by the proximal policy optimization algorithm, and excessive parameter updates are truncated. The parameter update is limited to a certain range, which reduces the oscillation problem in the process of gradient descent, and realizes quick convergence of the training process. The TCP

收稿日期：2020-09-29。 作者简介：王龙翔(1988—)，男，工程师；董小社(通信作者)，男，教授，博士生导师。 基金项目：国家重点研发计划资助项目(2018YFB0203902)。
网络出版时间：2021-01-11 网络出版地址：http://kns.cnki.net/kcms/detail/61.1069.T.20210108.1812.004.html

congestion control algorithm based on proximal policy optimization is implemented on NS3 simulator, and compared with the mainstream congestion control algorithms such as cubic, HighSpeed and NewReno. The results show that the throughput performance of TCP-PPO2 can reach more than 2 — 3 times of the comparison method, while the delay value of 80% of the sampling points only increases 4% compared with the minimum link delay.

Keywords: virtual data space; proximal policy optimization; congestion control; TCP

当前,国家高性能计算环境中存储资源广域分散且隔离自治,大型计算应用迫切需要可支持跨域统一访问、广域数据共享、存储与计算协同的全局数据空间。因此,我国拟构建跨域虚拟数据空间,实现广域安全可靠数据共享、计算与存储高效协同、跨域多源数据聚合处理等关键科学问题,从而发挥广域资源聚合效应,有效支撑大型计算应用。虚拟数据空间的部署环境包括 3 个国家级超算中心(广州、济南、长沙)、两个国家网格南北主节点(中国科学院、上海)。虚拟数据空间在线存储近 30 PB,活跃用户数超过6 000个,长期支撑数值模拟、大数据、人工智能等众多大型计算应用。虚拟数据空间存储数据规模达到 PB 级,其上层典型应用包括天气预报、全基因组关联分析等。不同超算中心在进行跨域节点数据迁移时,规模通常可达 GB 级甚至 TB 级,对网络传输性能提出了挑战。

为了实现虚拟数据空间可靠数据迁移,需要构建高效的可靠网络传输协议,而拥塞控制是实现高效可靠传输的关键技术。虚拟数据空间构建于广域网之上,其网络环境复杂多变,尽管在过去 30 年中研究者提出了各种各样的 TCP 拥塞控制算法(例如 NewReno、Cubic 等),但是这些算法普遍针对特定的网络环境,只能按照预先定义的规则进行拥塞控制,难以适应虚拟数据空间复杂多变的网络环境。

NewReno[1] 和 Cubic[2] 使用数据包丢失来检测拥塞,并在检测到拥塞后降低拥塞窗口长度。Vegas[3] 使用延迟、而不是丢包作为拥塞信号,可以解决基于丢包的拥塞控制问题。当 Vegas 检测到往返时延(RTT)超过设定值时,就会开始降低拥塞窗口长度。Westwood[4] 改良自 NewReno,基于传输能力的拥塞控制机制使用链路发送能力的预测作为拥塞控制的依据,通过测量确认字符(ACK)包来确定合适的发送速度,并以此调整窗口和慢启动阈值。混合拥塞控制机制组合两种拥塞控制机制,以得到它们各自的优势,进而更好地进行拥塞控制。Compound[5]、BBR[6] 都属于混合拥塞控制机制。

传统拥塞控制机制使用确定的规则集对拥塞窗口及其他相关参数进行控制,很难适应现代网络的复杂性和快速发展。因此,研究者提出了基于强化学习的拥塞控制算法。强化学习作为机器学习的研究热点,已经广泛应用于无人机控制[7]、机器人控制[8]、优化与调度[9-11] 以及游戏博弈[12] 等领域。强化学习的基本思想是构造一个智能体,使智能体与环境进行互动,通过最大化智能体从环境中获得的累计奖赏,学习到完成目标的最优策略。相比传统拥塞控制算法,基于强化学习的拥塞控制算法适应性好,能自主从网络环境中学习新的拥塞控制策略。

文献[13]提出了一种基于强化学习的算法生成拥塞控制规则,专门针对多媒体应用优化体验质量。文献[14]使用强化学习算法来自适应地更改参数配置,从而提高了视频流的体验质量。文献[15]提出了一种自定义的拥塞控制算法 Hd-TCP,应用深度强化学习从传输层角度处理高铁上网络频繁切换引起的网络体验较差的情况。文献[16]利用模型辅助的深度强化学习框架提高了虚拟网络功能的适用性。文献[17]主要针对灾难性 5G 毫米波网络,通过监测节点的移动性信息和信号强度,并通过预测何时断开和重新连接网络来调整 TCP 拥塞窗口长度。文献[18]提出了一种基于深度学习的 5G 移动边缘计算拥塞窗口长度。文献[19]基于深度强化学习,设计并开发了一种针对命名数据网络的拥塞控制机制 DRL-CCP。TCP-Drinc[20] 是基于深度强化学习的无模型智能拥塞控制算法,它从过去的网络状态和经验中获得特征值,并根据这些特征值的集合调整拥塞窗口长度。TCP-Drinc 在吞吐量和 RTT 之间取得了平衡,比 NewReno、Vegas 等算法具有更稳定、平均的表现,但在吞吐量上并没有明显的改善。Rax 算法[21] 使用在线强化学习,根据给定的奖励函数和网络状况维持最佳的拥塞窗口长度。该算法丢包率较低,但对比 Reno、PCC 等算法,吞吐率提升较小。QTCP[22] 基于 Q-learning 进行拥塞控制[23],吞吐率有进一步提升。Q-learning 的核心思想是求出所有状态-动作对(s,a)的价值 Q,Q 代表了在当前状态 s 下选择 a 可以获得的回合内预期

奖励值。如果求出了所有状态-动作对(s,a)的价值Q，则只需每次在状态s下选择能使Q最大的动作a即可实现最优拥塞控制策略。然而，Q-learning 算法存在学习速度慢、收敛难的问题。由于 Q-learning 算法旨在求出所有状态-动作对(s,a)的无偏 Q，因此需要根据 Bellman 方程反复迭代才能求出 Q 的准确值，当 Q 发生轻微变化时，可能导致训练过程发生反复振荡。当 Q-learning 算法的动作空间较大时，Q-learning 极易收敛到局部最优解，而基于策略梯度的强化学习算法则解决了 Q-learning 算法存在的学习速度慢、收敛难等缺陷，策略梯度算法的思想是直接优化策略函数，通过梯度上升的方式使策略函数获得的奖励值最大。近端策略优化（PPO2）是目前最佳的策略梯度算法之一[24]，已被 OpenAI 公司作为默认梯度策略算法。

有鉴于此，本文提出了基于 PPO2 算法的拥塞控制算法 TCP-PPO2，该算法可以在学习过程快速收敛，实现虚拟数据空间的高效可靠数据迁移。与主流拥塞控制算法相比的结果表明，本文算法在虚拟数据空间应用环境中可行有效。

1　基于 PPO2 的 TCP 拥塞控制算法

TCP-PPO2 算法框架如图 1 所示。强化学习需要构造环境和智能体。将虚拟数据空间网络作为环境，通过观察环境中的状态信息，构造智能体使用的策略函数，生成最优控制动作，策略函数采用人工神经网络进行拟合。智能体根据策略函数输出的动作对拥塞窗口长度进行调节，优化虚拟数据空间网络性能。在生成动作并与环境互动后，智能体会从环境中收获奖励值。智能体根据奖励值评判所选动作的优劣，并根据奖励值更新人工神经网络参数，使策略函数能够生成收获奖励值更多的动作。

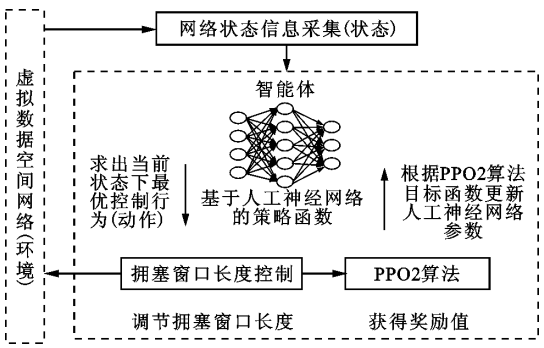


图 1　TCP-PPO2 算法框架

Fig. 1　Framework of TCP-PPO2 algorithm

根据是否求出状态概率转移矩阵，可将强化学习分为无模型和基于模型两种类型。

基于模型算法从环境模型中交互得到样本，根据样本估计状态概率转移矩阵对环境进行建模。获得的样本能够多次使用，样本利用率高。根据状态概率转移矩阵能够更好地设计奖励值来引导智能体学习。但是，基于模型算法对环境的建模可能存在偏差。模型一旦确立，训练好之后，环境出现新的改变就会失效，泛化能力差。基于模型算法的典型代表是动态规划。

无模型算法直接根据从环境交互中得到的反馈信息（奖励值）求出最优控制策略，而不是求出状态概率转移矩阵。该算法泛化能力强，但是存在学习效率低、收敛慢的问题。这是因为该算法类似将环境作为一个黑盒进行反复试错求出最优控制策略，智能体缺少足够的指引。Q-learning[23]、PPO2[24]都是典型的无模型算法。

这两类算法的区别在于是否能够求出状态概率转移矩阵。对于本文要研究的虚拟数据空间网络拥塞控制，求出状态概率转移矩阵难度大、代价高，而且网络环境会不断发生变化，从而导致需要不断更新状态概率转移矩阵。因此，本文采用的是无模型算法，智能体在不了解状态概率转移矩阵的情况下求得最优拥塞控制策略。

1.1　问题形式化

将基于强化学习的 TCP 拥塞控制过程抽象为一个可部分观察的马尔可夫决策过程，定义为五元组$\{S,A,R,\boldsymbol{P},\gamma\}$。其中：$S$ 为所有环境状态的集合，$s_t\in S$ 表示在 t 时刻观察到的状态，初始状态为 s_0；A 为可执行动作的集合，$a_t\in A$ 表示在 t 时刻所采取的动作；R 为奖励值函数，定义为 $R(s_t,a_t)=E[R_{t+1}\mid s_t,a_t]$，表示在 t 时刻观察到状态为 s_t、选择动作 a_t 后，在 $t+1$ 时刻收到奖励 R_{t+1}；$\boldsymbol{P}$ 为转移概率矩阵；$\gamma\in[0,1]$为折扣因子，是对未来得到奖励的惩罚比例，折扣因子体现了强化学习算法的设计思想，即优先考虑能够立刻得到的奖励值，未来得到的奖励值会按一定比例进行衰减。

强化学习算法从初始状态 s_0 开始，根据当前观察到的状态 s_t，由策略函数 $\pi(a_t\mid s_t)$选择动作 a_t，根据状态转移概率 $\boldsymbol{P}(s_{t+1}\mid s_t,a_t)$到达新状态 s_{t+1}，从环境中得到奖励 r_{t+1}。强化学习的目标是优化策略函数使奖励期望值最大，奖励值期望定义为

$$\hat{R}=\sum_{t=0}^{T}\gamma^t r_t \tag{1}$$

式中 T 代表结束时刻。

1.2 PPO2 原理

强化学习算法需要设计策略函数 $\pi(a_t|s_t)$，使其能够在状态 s_t 下生成执行某个动作 a_t 的概率。人工神经网络理论上能够拟合任意函数，因此目前强化学习算法通过人工神经网络拟合策略函数 $\pi(a_t|s_t)$，神经网络参数记作 θ。强化学习的目标是使得每次做出的动作都能取得最大奖励值，核心是如何评判所选择动作的优劣。为此，定义优势函数

$$\hat{L}_{\pi_\theta,t}=\hat{R}_t-V_\phi(s_t) \tag{2}$$

式中 $V_\phi(s_t)$ 是状态 s_t 的值函数，反映了在状态 s_t 下，预期本次回合结束后能够取得的所有累计奖励值。优势函数反映了在时刻 t 选择动作 a_t 相对平均动作的优势。如果保存所有状态 s_t 和动作 a_t 对应的价值 v_t 为二维表格，由于状态 s_t 取值范围庞大，会导致二维表格存储空间巨大而难以存储。因此，同样选择人工神经网络对值函数 $V_\phi(s_t)$ 进行近似表示。最终，定义强化学习的优化目标函数

$$L^{\mathrm{MSE}}=E_{\pi_{\theta_k}}(\hat{R}_t-V_\phi(s_t))^2 \tag{3}$$

式(3)函数的目标是通过更新策略函数参数 θ 使得每次做出动作都能获得更大的奖励值。然而，目标函数 L^{MSE} 存在的问题是如果参数 θ 更新幅度过大，会造成梯度上升时反复振荡而无法快速收敛到最优点。为此，PPO2 算法重新定义目标函数

$$L^{\mathrm{clip}}(\theta)=\hat{E}_t[\min(r_t(\theta)\hat{A}_t,\mathrm{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_t)] \tag{4}$$

式中：clip 函数是截断函数，定义为

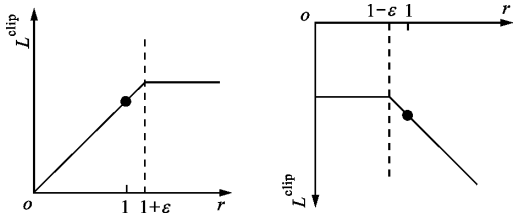
$$\mathrm{clip}(r,1-\varepsilon,1+\varepsilon)=\begin{cases}r, & 1-\varepsilon<r<1+\varepsilon\\ 1-\varepsilon, & r\leqslant 1-\varepsilon\\ 1+\varepsilon, & r\geqslant 1+\varepsilon\end{cases} \tag{5}$$

$r_t(\theta)$ 为概率比函数，定义为

$$r_t(\theta)=\frac{\pi_\theta(a_t\mid s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t\mid s_t)} \tag{6}$$

$r_t(\theta)$ 反映了参数更新的变化幅度，$r_t(\theta)$ 越大，则更新参数幅度越大，反之则越小。

式(3)的目标是求得值函数 $V_\phi(s_t)$ 的有偏估计，因此采用常用的最小二乘法定义目标函数，平方运算保证了目标函数非负性。式(4)中的优势函数取值为正时，代表当前动作获取的奖励值高于平均值，目标函数优化目标是让智能体尽量选择这类动作；优势函数为负时，代表当前动作获取的奖励值低于平均值，智能体应该避免选择该动作。$L^{\mathrm{clip}}(\theta)$ 函数通过截取 $r_t(\theta)$，将其限制在 $[1-\varepsilon,1+\varepsilon]$ 之间，从而避免更新波动过大。$L^{\mathrm{clip}}(\theta)$ 函数示意如图 2 所示。当优势函数 $L>0$ 时，如果 $r_t(\theta)$ 大于 $1+\varepsilon$，则将其截断，使其不会过大。同样，当 $L<0$ 时，如果 $r_t(\theta)$ 小于 $1-\varepsilon$，也将其截断，使其不会过小。$L^{\mathrm{clip}}(\theta)$ 函数保证了 $r_t(\theta)$ 不会出现剧烈波动。



(a)$L>0$ (b)$L<0$

图 2 截断函数示意

Fig. 2 Schematic diagram of clip function

1.3 算法收敛性

PPO2 在 TRPO 算法[24]基础上进一步改进，两者都是基于 minorize-maximization 算法，目标是最大化期望奖励 $\eta(\theta^*)$。其中，η 为折扣奖励函数，θ^* 为待寻找的最佳策略参数。在每一次迭代中，找到一个替代函数 M，M 为折扣期望奖励的下界，也是当前策略下对折扣期望奖励的估计。本文 M 为目标函数 $L^{\mathrm{clip}}(\theta)$，其迭代过程如图 3 所示。

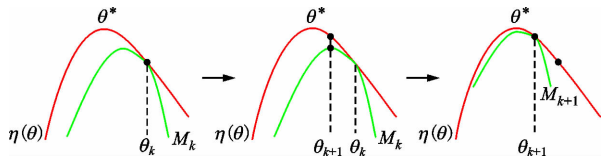


图 3 PPO2 迭代过程示意

Fig. 3 Schematic diagram of PPO2 iteration process

当前策略参数 θ_k 建立折扣奖励函数 η 的下界 M_k。最优化 M_k，找到 θ_{k+1} 作为下一个策略参数。用 θ_{k+1} 重新估计下界 M_{k+1}，并重复这个过程。由于只有有限个可能的策略，且每一次迭代的策略都使得新策略更加接近最佳策略，PPO2 最终会收敛到局部或全局最优。

为了对这一过程进行证明，定义折扣奖励函数

$$\eta(\pi)=E\Big[\sum_{t=0}^{\infty}\gamma^t r(s_t)\Big] \tag{7}$$

折扣奖励函数是强化学习算法要优化的目标函数。

定义函数

$$L_\pi(\tilde{\pi})=\eta(\pi)+\sum_s\rho_\pi(s)\sum_a\tilde{\pi}(a\mid s)A_\pi(s,a) \tag{8}$$

式中 $\rho_\pi(s)$ 是状态分布，公式为

$$\rho_\pi(s)=P(s_0=s)+\gamma P(s_1=s)+\gamma^2P(s_2=s)+\cdots \tag{9}$$

文献[25]证明了不等式(10)成立

$$\eta(\tilde{\pi})\geqslant L_\pi(\tilde{\pi})-CD_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi}) \tag{10}$$

式中

$$C=\frac{4\varepsilon\gamma}{(1-\gamma)^2} \tag{11}$$

$$\left.\begin{aligned} D_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi})&=\max_s D_{\mathrm{KL}}(\pi(\cdot\mid s)\,\|\,\tilde{\pi}(\cdot\mid s))\\ \varepsilon&=\max_{s,a}\mid A_\pi(s,a)\mid \end{aligned}\right\} \tag{12}$$

其中 D_{KL} 是两个策略之间的 KL 散度。

定义替代函数 M 为

$$M_i(\pi)=L_{\pi_i}(\pi)-CD_{\mathrm{KL}}^{\max}(\pi_i,\pi) \tag{13}$$

根据式(10)定义有

$$\eta(\pi_{i+1})\geqslant M_i(\pi_{i+1}) \tag{14}$$

由于两个相同的策略 KL 散度为 0，因此

$$\eta(\pi_i)=M_i(\pi_i)=L_i(\pi_i) \tag{15}$$

从而得到

$$\eta(\pi_{i+1})-\eta(\pi_i)\geqslant M_i(\pi_{i+1})-M_i(\pi_i) \tag{16}$$

如果新的策略函数 π_{i+1} 能使得 M_i 最优，那么有不等式 $M_i(\pi_{i+1})-M_i(\pi_i)\geqslant 0$ 成立，进而有

$$\eta(\pi_{i+1})-\eta(\pi_i)\geqslant 0 \tag{17}$$

因此，只要不断寻找能使 M_i 最优的策略就能保证强化学习目标函数 η 在每次迭代中不会下降，最终收敛到局部或者全局最优点，即

$$\max_\theta L_{\pi_{\theta_{\mathrm{old}}}}(\pi)-CD_{\mathrm{KL}}^{\max}(\pi_{\theta_{\mathrm{old}}},\pi_\theta) \tag{18}$$

文献[25]指出，式(18)更新幅度过小，导致收敛慢。为增加策略更新幅度，可将优化问题转换为

$$\max_\theta L_{\pi_{\theta_{\mathrm{old}}}}(\pi),\quad \text{s.t. } D_{\mathrm{KL}}^{\max}(\pi_{\theta_{\mathrm{old}}},\pi_\theta)\leqslant\delta \tag{19}$$

由于约束条件中的最大 KL 散度 $D_{\mathrm{KL}}^{\max}(\pi_{\theta_{\mathrm{old}}},\pi_\theta)$ 有无数多个，在实际中不可解。文献[25]指出可用平均散度近似最大散度。因此，约束条件变为

$$\overline{D}_{\mathrm{KL}}^{\theta_{\mathrm{old}}}(\pi_{\theta_{\mathrm{old}}},\pi_\theta)\leqslant\delta \tag{20}$$

新的优化问题为

$$\max_\theta L_{\pi_{\theta_{\mathrm{old}}}}(\pi),\quad \text{s.t. } \overline{D}_{\mathrm{KL}}^{\theta_{\mathrm{old}}}(\pi_{\theta_{\mathrm{old}}},\pi_\theta)\leqslant\delta \tag{21}$$

对 $L_{\pi_{\theta_{\mathrm{old}}}}(\pi)$ 展开，并采用重要性采样进行替换，可将优化目标变化为

$$\left.\begin{aligned} &\max_\theta \hat{E}_t\left[\frac{\pi_\theta(a_t\mid s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t\mid s_t)}\hat{A}_t\right]\\ &\text{s.t. }\hat{E}_t[D_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}},\pi_\theta)]\leqslant\delta \end{aligned}\right\} \tag{22}$$

论述 PPO2 算法的文献[24]指出，为了算法更加易于实现，可将优化目标函数 L 变为

$$\hat{E}_t[\min(r_t(\theta)\hat{A}_t,\mathrm{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_t)] \tag{23}$$

优化问题变为

$$\max_\theta \hat{E}_t[\min(r_t(\theta)\hat{A}_t,\mathrm{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_t)] \tag{24}$$

式(24)仍然满足不等式(16)，因此 PPO2 可以最终收敛到最优点。

1.4　状态空间设计

选取合理的状态 s_t 是实现高效强化学习算法的关键，只有观察到足够多的信息才能使强化学习算法做出正确的动作选择。然而，状态信息过多也会增加计算量，减慢学习速度。因此，本文参考了 Cubic 等主流 TCP 算法进行决策需要的状态参数，设计状态 s_t。s_t 包含以下参数。

(1)当前相对时间 t_{r}。定义为从 TCP 建立连接开始到目前已消耗的时间。在 Cubic 等算法中，窗口长度被设计为时间 t_{r} 的 3 次函数。因此，t_{r} 是决定拥塞窗口的重要参数。

(2)当前拥塞窗口长度。拥塞控制算法需要根据当前拥塞窗口长度来调节窗口新值，如果当前拥塞窗口长度较小，则可以更快的速率增加窗口长度，如果窗口较大，则停止增加窗口或更缓慢地增加窗口长度。

(3)未被确认的字节数。定义为已发送但还未被接收方确认的字节数。如果把网络链路比喻做水管，则未被确认的字节数可以形象地理解为管道中储存的水量。该参数也是拥塞控制算法需要参考的重要参数，如果管道中水量充足，则应该停止或减少向管道中注水，如果管道中水量较小，则应该向管道中增加注水量，并且可以根据管道中的水量决定注水速率(拥塞窗口长度)。

(4)已收到的 ACK 包数量。该参数能够间接反映拥塞情况，如果收到的 ACK 包数量正常，则说明网络状况良好，未发生拥塞，可以适时增大拥塞窗口长度，否则说明网络发生拥塞，应该维持或减小拥塞窗口长度。

(5)RTT。时延指一个数据包从发送到接收确认包花费的总时间，可形象地理解为数据从发送端到接收端进行一次往返的时间。时延跟网络拥塞情况密切相关，如果网络拥塞严重，则时延会显著上升。因此，时延可以反映网络拥塞情况，拥塞控制算法可以根据时延对拥塞窗口进行调节。

(6)吞吐率。定义为接收方每秒确认的数据字节数。该参数直接反映了网络状况，吞吐率高说明目前链路中已发送足够的数据包，否则说明当前网络带宽剩余较多，可向链路中增加发送数据包。

(7)丢失包数量。丢失包数量越多说明当前网络拥塞严重，需要减小拥塞窗口长度，丢失包数量少说明当前网络未发生拥塞，应该增大拥塞窗口长度。

1.5 动作空间设计

a_t 为在时刻 t 对拥塞窗口做出的控制动作。本文定义动作为将拥塞窗口长度 c 增加 n 个段长度 s'

$$c = c_{\text{old}} + ns' \tag{25}$$

式(25)设计的思路是提供一个泛化公式，根据观察到的状态参数信息，决定拥塞窗口长度增长速率。在不同的网络场景下，选择不同的策略。在高带宽环境下，调节 $n>1$，使拥塞窗口长度以指数速度增长；在低带宽环境下，调节 $n=1$，使拥塞窗口以线性速度增长；在网络发生拥塞时，调节 $n\leqslant 0$，保持或减小拥塞窗口长度，减轻网络拥塞压力。

1.6 奖励函数

奖励 r_t 定义为在时刻 t 从环境中收到的奖励，设计奖励函为

$$r_t = \alpha\left(\frac{O}{O_{\max}}\right) - (1-\alpha)\frac{l_{\min}}{l} \tag{26}$$

式中：O 为当前观察到的吞吐率，$O_{\max}$ 为历史观察到的最大吞吐率，两者的比反映了动作 a_t 能增加的吞吐率效果；l 代表观察期间的平均时延，$l_{\min}$ 代表历史中观察到的最小时延，两者的比反映了动作 a_t 改善的时延效果；α 为权重因子，属于超参数，反映了吞吐率和时延对奖励的权重比例。α 决定了拥塞控制算法的优化目标更侧重于吞吐率还是时延。本文选择 $\alpha=0.5$ 以平衡吞吐率和时延。此外，保存历史最小吞吐率与最大时延。当观察到当前吞吐率小于等于最小吞吐率或者大于等于最大时延时，设置获得奖励为 -10，使智能体避免到达这两种极端状态。

1.7 算法描述及复杂度分析

算法的输入为网络当前状态 s_t，输出为新的窗口长度 c_{new}，伪代码如下。

输入：s_t={拥塞窗口长度，ACK 包数量，时延，吞吐率，丢包率}

输出：调节后新拥塞窗口长度

1. 初始化策略参数 $\theta_0=\theta_{\text{old}}=\theta_{\text{new}}$
2. 运行策略 π_{θ_k} 共 T 个时间步，收集{s_t，a_t}
3. $\theta_{\text{old}}\leftarrow\theta_{\text{new}}$
4. $r_t=\alpha\left(\frac{O}{O_{\max}}\right)-(1-\alpha)\frac{l_{\min}}{l}$
5. $\hat{R}=\sum_{t=0}^{T}\gamma^t r_t$
6. $L^{\text{clip}}(\theta)=\hat{E}_t[\min(r_t(\theta)\hat{A}_t,\text{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_t)]$
7. 通过梯度上升法更新参数 θ，使 $L^{\text{clip}}(\theta)$ 最大
8. $c=c_{\text{old}}+ns'$

TCP-PPO2 只需存储神经网络的参数，本文实验中构建了一个 3 层神经网络，因此空间复杂度为 $O(1)$。TCP-PPO2 在做训练和推理时需要根据输入的观察状态由模型计算得到动作值，因此时间复杂度与输入数据量成正比，即 $O(n)$。

2 实 验

2.1 实验环境

2.1.1 *软硬件环境* 实验使用了一台高性能服务器，具体配置如下：①CPU，Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz；②内存，32 GB DDR4；③GPU，NVIDIA Titan V；④操作系统，Red Hat 4.8.5-28。

通过 NS3 仿真器模拟了虚拟数据空间网络拓扑结构，并实现了 TCP-PPO2 算法，与具有代表性的 TCP 拥塞控制算法 Cubic、NewReno 和 HighSpeed 进行了对比。Cubic 在 Linux 内核 2.6.19 版本以后作为默认 TCP 拥塞控制算法；NewReno 是经典的拥塞控制算法；HighSpeed 是面向高速网络环境设计的拥塞控制算法。TCP-PPO2 时间步长设为 0.1 s，一共训练了 50 万步，在训练到 6 万步以后，获得的奖励值已趋于稳定，表明算法已经收敛。

2.1.2 *网络拓扑* 实验用经典的哑铃型网络拓扑结构模拟了虚拟数据空间两个超算中心中间的网络特点。网络拓扑如图 4 所示。图中：N1 和 N2 代表两个超算中心之间的前端通信节点，负责进行虚拟数据空间网络数据迁移，N1 为数据发送方，N2 为数据接收方；N1-T1 和 N2-T2 链路代表超算中心内部网络链路，平均网络带宽设置为 1 Gb/s，时延为 60 μs，丢包率为 0；T1-T2 代表广域网通信链路，实验中设置平均网络带宽为 100 Mb/s，时延为 80 ms，丢包率设为 10^4。这些参数设置来源于虚拟数据空间广域网环境性能实测数据，尽量模拟了真实广域网特点。

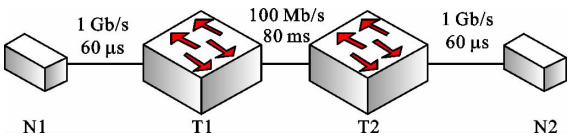


图 4 网络拓扑

Fig. 4 Network topology

2.1.3 *PPO2 参数设置* PPO2 的主要参数设置如下：折扣因子为 0.99，学习速率为 0.000 25，ε 为 0.2，每次更新运行的训练步数为 128。

2.2　吞吐率对比

吞吐率为每秒确认的全部数据包数量。图 5 是吞吐率性能对比，可以看出，TCP-PPO2 的网络吞吐率约为 HighSpeed 和 Cubic 算法的 2 倍，约为 NewReno 算法的 3 倍。图 6 是 4 种算法吞吐率的累计概率密度函数曲线对比，可以看出：NewReno 只有约 3%采样点的吞吐率大于 4 MB/s，1%采样点的吞吐率大于 6 MB/s；Highspeed 和 Cubic 有 30%采样点的吞吐率大于 4 MB/s，约 3%采样点的吞吐率大于 6 MB/s；TCP-PPO2 有 90%采样点的吞吐率大于 4 MB/s，40%采样点的吞吐率大于 6 MB/s。结果表明：NewReno 这类传统拥塞控制算法已无法适应虚拟数据空间的广域网特点，不适合应用于虚拟数据空间数据迁移；Cubic 和 Highspeed 比 NewReno 具有显著的性能提升，但是仍未能完全有效利用可用带宽实现高速传输；TCP-PPO2 具有最好的性能，在进行一定的学习后，能够充分利用网络带宽实现虚拟数据空间高效数据迁移。

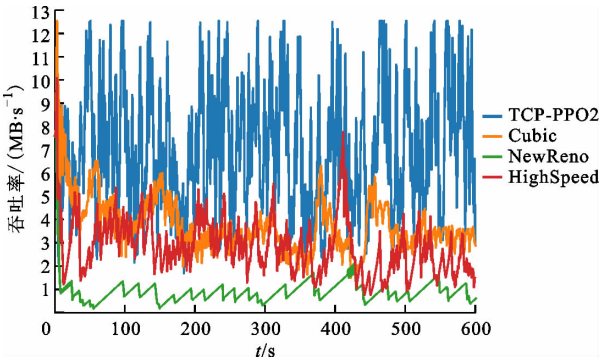


图 5　吞吐率性能对比

Fig. 5　Comparison of throughput performance

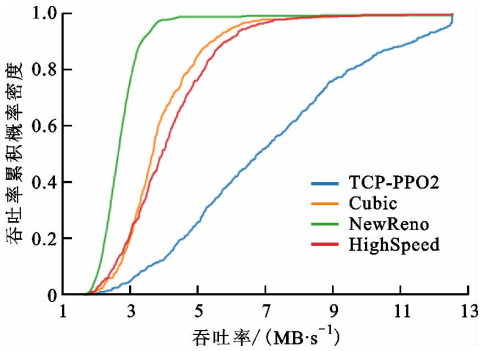


图 6　吞吐率累计概率密度分布对比

Fig. 6　Comparison of cumulative probability density distribution of throughput rate

2.3　网络时延对比

RTT 代表一个数据包从发送到接收到确认包的耗费时间，反映了当前网络延迟状况。图 7 是 RTT 对比，可以看出，总体上 TCP-PPO2 算法的 RTT 相比其他 3 种算法的有所上升，这是由于 TCP-PPO2 算法更加激进，尝试利用所有可用带宽，向链路中发送过多数据包，造成网络拥塞，导致 RTT 增加，但是 TCP-PPO2 算法的 RTT 相比其他 3 种算法的上升幅度不大。图 8 是 RTT 累计概率密度函数对比，可以看出，TCP-PPO2 算法有 80%的 RTT 小于 167 ms。由于链路本身 RTT 最小值为 160 ms，因此 TCP-PPO2 算法的大部分 RTT 相比最小值只增加了 4%。

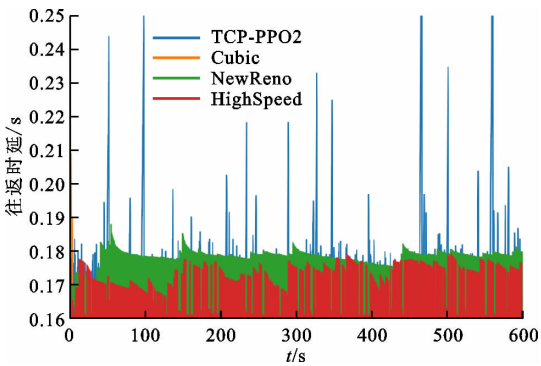


图 7　RTT 对比

Fig. 7　Comparison of RTT

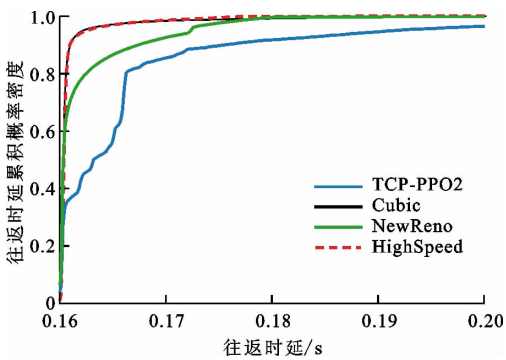


图 8　RTT 累计概率密度分布对比

Fig. 8　Comparison of RTT cumulative probability density distribution

2.4　队列长度对比

对 T1 上的队列长度进行了采样，结果如图 9 所示。可以看出，TCP-PPO2 算法的队列长度显著高于其他 3 种算法的。这是因为 TCP-PPO2 算法单位时间内发送的数据包数量最多，所以在 T1 路由器上需要缓存的队列长度也最长。

2.5　丢包率对比

N1 向 N2 发送数据过程中的丢包率如图 10 所示。可以看出：4 种算法的丢包率都接近 0.01%，与

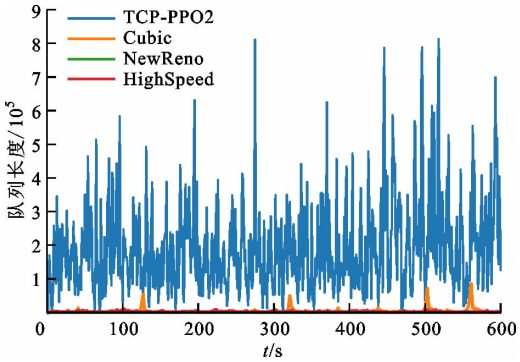


图 9 队列长度对比

Fig. 9 Comparison of queue length

NS3 参数设置一致；TCP-PPO2 丢包率为 0.124%，略高于其他 3 种算法的。这是因为 TCP-PPO2 发送的数据包最多，部分数据包由于链路节点缓存已满被丢弃，从而出现丢包现象。

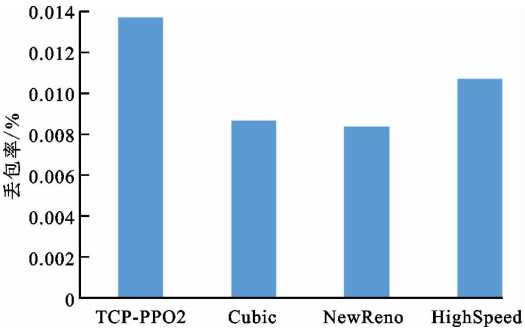


图 10 丢包率对比

Fig. 10 Comparison of packet loss rate

2.6 收敛速度对比

DQN 算法是 *Q*-learning 家族的最新研究成果，采用神经网络对 *Q* 表格进行了近似，已应用在 Alpha Go 智能围棋系统中。本文对 PPO2 算法和 DQN 的收敛速度进行了对比，结果如图 11 所示。可以看出，PPO2 算法在训练到 7 万步以后，收到的奖励值已趋于稳定，表明算法已经收敛。根据 1.3 小节的收敛性分析，PPO2 具有单调上升性，图 11 验证了该结论，PPO2 收到的奖励值随训练步数不断上升，最终趋于稳定。DQN 算法在训练到 42 万步以后仍然反复振荡，难以收敛。实验结果表明 PPO2 算法具有更快的收敛速度。

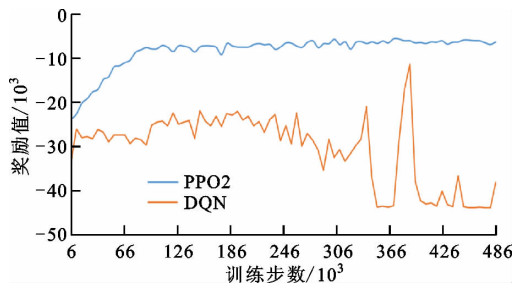


图 11 收敛速度对比

Fig. 11 Comparison of convergence speed

3 结 论

虚拟数据空间对于聚合国家高性能计算资源具有重要意义，高效可靠数据传输是构建虚拟数据空间的核心技术。本文针对主流 TCP 拥塞控制算法适应性差、无法有效利用虚拟数据空间网络带宽等问题，提出了一种基于近端策略优化算法的 TCP 拥塞控制算法，用于实现虚拟数据空间高效可靠数据迁移。本文得出的主要结论如下。

(1)提出了基于近端策略优化算法的 TCP 拥塞控制算法，将基于强化学习的 TCP 拥塞控制过程抽象为可部分观察的马尔可夫决策过程。通过借鉴主流算法，合理设计了状态空间、动作空间、奖励函数。

(2)通过 NS3 仿真实验对比得出结论，TCP-PPO2 与 HighSpeed、Cubic、NewReno 算法相比吞吐率可达 2～3 倍以上。

未来将在真实虚拟数据空间系统中测试 TCP-PPO2 算法的性能，并针对测试性能结果，进一步提出优化算法，更好地服务国家高性能计算环境。

参考文献：

[1] FLOYD S, HENDERSON T. The NewReno modification to TCP's fast recovery algorithm: RFC2582 [A/OL]. [2020-07-01]. https://dl.acm.org/doi/pdf/10.17487/RFC2582.

[2] HA S, RHEE I, XU Lisong. Cubic: a new TCP-friendly high-speed TCP variant [J]. ACM SIGOPS Operating Systems Review, 2008, 42(5): 64-74.

[3] BRAKMO L S, O'MALLEY S W, PETERSON L L. TCP Vegas: new techniques for congestion detection and avoidance [C]// Proceedings of the Conference on Communications Architectures, Protocols and Applications. New York, USA: ACM, 1994: 24-35.

[4] MASCOLO S, CASETTI C, GERLA M, et al. TCP westwood: bandwidth estimation for enhanced transport over wireless links [C]// Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (MOBICOM). New York, USA: ACM. New York, USA: ACM, 2001: 287-297.

[5] TAN K, SONG J, ZHANG Q, et al. A compound TCP approach for high-speed and long distance networks [C]// Proceedings of the 25th IEEE International Conference on Computer Communications (INFO-

COM). Piscataway, NJ, USA: IEEE, 2006: 4146841.

[6] CARDWELL N, CHENG Y, GUNN C S, et al. BBR: congestion-based congestion control [J]. Communications of the ACM, 2017, 60(2): 58-66.

[7] XIAO Liang, LU Xiaozhen, XU Dongjin, et al. UAV relay in VANETs against smart jamming with reinforcement learning [J]. IEEE Transactions on Vehicular Technology, 2018, 67(5): 4087-4097.

[8] NIROUI F, ZHANG Kaicheng, KASHINO Z, et al. Deep reinforcement learning robot for search and rescue applications: exploration in unknown cluttered environments [J]. IEEE Robotics and Automation Letters, 2019, 4(2): 610-617.

[9] HUANG Shanfeng, LV B, WANG Rui, et al. Scheduling for mobile edge computing with random user arrivals: an approximate MDP and reinforcement learning approach [J]. IEEE Transactions on Vehicular Technology, 2020, 69(7): 7735-7750.

[10] CAO Zhengcai, LIN Chengran, ZHOU Mengchu, et al. Scheduling semiconductor testing facility by using cuckoo search algorithm with reinforcement learning and surrogate modeling [J]. IEEE Transactions on Automation Science and Engineering, 2019, 16(2): 825-837.

[11] 颛孙少帅，杨俊安，刘辉，等. 采用双层强化学习的干扰决策算法 [J]. 西安交通大学学报，2018，52(2)：63-69.
ZHUANSUN Shaoshuai, YANG Junan, LIU Hui, et al. An algorithm for jamming decision using dual reinforcement learning [J]. Journal of Xi'an Jiaotong University, 2018, 52(2): 63-69.

[12] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518(7540): 529-533.

[13] HABACHI O, SHIANG H P, VAN DER SCHAAR M, et al. Online learning based congestion control for adaptive multimedia transmission [J]. IEEE Transactions on Signal Processing, 2013, 61(6): 1460-1469.

[14] VAN DER HOOFT J, PETRANGELI S, CLAEYS M, et al. A learning-based algorithm for improved bandwidth-awareness of adaptive streaming clients [C] // Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). Piscataway, NJ, USA: IEEE, 2015: 131-138.

[15] CUI Laizhong, YUAN Zuxian, MING Zhongxing, et al. Improving the congestion control performance for mobile networks in high-speed railway via deep reinforcement learning [J]. IEEE Transactions on Vehicular Technology, 2020, 69(6): 5864-5875.

[16] GU Lin, ZENG Deze, LI Wei, et al. Intelligent VNF orchestration and flow scheduling via model-assisted deep reinforcement learning [J]. IEEE Journal on Selected Areas in Communications, 2020, 38(2): 279-291.

[17] NA W, BAE B, CHO S, et al. DL-TCP: Deep learning-based transmission control protocol for disaster 5G mmwave networks [J]. IEEE Access, 2019, 7: 145134-145144.

[18] XIE Ruitao, JIA Xiaohua, WU Kaishun. Adaptive online decision method for initial congestion window in 5G mobile edge computing using deep reinforcement learning [J]. IEEE Journal on Selected Areas in Communications, 2020, 38(2): 389-403.

[19] LAN Dehao, TAN Xiaobin, LV Jinyang, et al. A deep reinforcement learning based congestion control mechanism for NDN [C] // Proceedings of the 2019 IEEE International Conference on Communications (ICC). Piscataway, NJ, USA: IEEE, 2019: 8761737.

[20] XIAO Kefan, MAO Shiwen, TUGNAIT J K. TCP-drinc: smart congestion control based on deep reinforcement learning [J]. IEEE Access, 2019, 7: 11892-11904.

[21] BACHL M, ZSEBY T, FABINI J. Rax: deep reinforcement learning for congestion control [C] // Proceedings of the 2019 IEEE International Conference on Communications (ICC). Piscataway, NJ, USA: IEEE, 2019: 8761187.

[22] LI Wei, ZHOU Fan, CHOWDHURY K R, et al. QTCP: adaptive congestion control with reinforcement learning [J]. IEEE Transactions on Network Science and Engineering, 2019, 6(3): 445-458.

[23] WATKINS C J C H, DAYAN P. *Q*-learning [J]. Machine Learning, 1992, 8(3): 279-292.

[24] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms [EB/OL]. [2020-07-01]. https: // arxiv. org/pdf/1707. 06347. pdf.

[25] SCHULMAN J, LEVINE S, MORITZ P, et al. Trust region policy optimization [C] // Proceedings of the 32nd International Conference on Machine Learning. Princeton, NJ, USA: International Machine Learning Society (IMLS), 2015: 1889-1897.

（编辑　陶晴）