# 基于 SARSA 学习算法的 USB 块传输研究*

张秋云[1]，江 虹[2]
（西南科技大学信息工程学院，四川 绵阳 621010）

**摘 要**：目前 USB 在 PC 外设中应用越来越多，传输数据量不断增加对 USB 传输效率要求越来越高。但实际应用中因 USB 系统软件、设备自身特性等因素的影响，使得数据传输过程中 USB 带宽资源浪费严重。针对该问题，利用 SARSA 学习算法设计一种 USB 块传输事务调度方法，根据当前状态智能的分配每一帧中的事务。仿真结果表明，在多种块传输情况下，该方法与系统方式相比明显提高了 USB 带宽有效利用率和吞吐量。

**关键词**：USB；SARSA；块传输；USB 带宽有效利用率
**中图分类号**：TP391.9 **文献标志码**：A **文章编号**：0529－6579（2014）05－0073－06

# USB Bulk Transfer Research Based on SARSA Learning Algorithm

*ZHANG Qiuyun*, *JIANG Hong*
(School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, China)

**Abstract**: USB is widely used in PC peripherals, and the increase of transmission data results in increasingly high requirement on transmission efficiency. However, there is a serious waste of USB bandwidth resources, due to the USB system software and characteristics of devices. A USB bulk transfer transaction scheduling method with SARSA learning algorithm for this problem, intelligently allocating the transactions in each frame based on current environment is designed. Simulation results show that this method significantly improves the utilization effectiveness of USB bandwidth and the throughput, compared to the method of the USB system, in the case of multiple bulk transfers.

**Key words**: USB; SARSA; bulk transfer; utilization effectiveness of USB bandwidth

USB 总线上的数据传输分为同步、中断、控制、块传输等四种类型[1]，不同设备采用的传输方式不尽相同，在有限的带宽资源下如何协调数据传输对实现 USB 高效传输非常重要。目前仅文献[2]针对同步传输的设备设计并实现了一种实时的 USB 驱动，使设备获得了可靠的 QoS 保证，同时也提高了 USB 带宽利用率，但对于其他方式的传输尚未进行研究。大容量存储设备、大数据量的传输一般采用优先级较低的块传输方式[1]，但目前针对该传输方式仍采用文献[1，3]中所描述的 USB 软件系统预设的资源分配机制。根据文献[1，3]中对 USB 软件系统的描述，该机制是一种“先到先得，尽力而为”的工作模式，进行带宽资源分配时需要不断“试错”以满足预定的分配原则，不具备感知、学习 USB 总线传输状态，调整预定分配原则的能力，使得数据传输过程中带宽未能得到充分利用。

针对 USB 软件系统对带宽资源分配中存在的问题，设计一种具有学习能力的带宽分配方法，对提高大数据传输效率很有必要。本文结合 SARSA 强化学习算法，设计一种大数据传输下的智能带宽分配方法，该方法在行动—评价的过程中获得环境知识，通过不断的“试错”来改进行动方案以适应环境，最终达到提高 USB 带宽的有效利用率和吞吐量的目的。

* **收稿日期**：2013－11－25
**基金项目**：国防基础科研计划资助项目（B3120110005）
**作者简介**：张秋云（1988 年生），男；**研究方向**：通信、智能控制等；E-mail：zhangqiuyun123@163.com

# 1 USB 数据传输

## 1.1 资源分配

USB 传输的 4 种类型中，同步和中断方式属于周期的数据传输（Periodic），而控制和块方式属于非周期的传输（NonPeriodic）。周期传输可以分配高达 90% 的总线带宽；控制传输最高可分配 10% 的总线带宽；块传输以尽力而为的方式利用剩余的总线带宽[1]。每类传输在每 1ms 帧中共享 USB 总线带宽，如图 1 所示，图中 SOF 表示帧的开始。

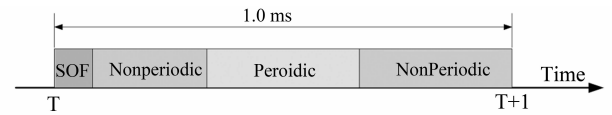



图 1 1 ms USB 帧中带宽预留分配

Fig. 1 The bandwidth reservation of the USB 1 ms time frame

数据传输时系统软件根据上述机制对当前传输类型进行资源分配，当传输同步、中断、控制对应业务量少或没有时，USB 系统软件可将剩余带宽资源尽量地分配给块传输[1]。

## 1.2 事务处理与数据帧

事务处理是 USB 总线数据传输的基本单位，主机和 USB 设备间的一次通信由一个或多个事务处理组成。总线驱动根据设备相关信息将 *IRP*（I/O Request Packet）转化生成相应的传输事务，主控制器驱动产生与每个传输事务相对应的传输描述符 *TD*（Transfer Descriptors）[4]。主控制器驱动按照“先到先得”和式（1）所述的原则[2-3]，选择 *TD*（即事务调度）组成数据帧列表，USB 主控制器依次读取处理该帧列表以完成数据传输。

$$\begin{cases} \sum_{i=1}^{n} T_{TD(i)} \leqslant 1(\text{ms}) \\ \text{SOF} + \sum_{i=1}^{n} \text{Sizeof}(D_{TD(i)} + P_{TD(i)}) \leqslant 1\,500(\text{Bytes}) \end{cases} \tag{1}$$

其中，$T_{TD(i)}$ 表示第 $i$ 个 *TD* 所需处理时间，SOF 为每一 USB 数据帧开始所需的字节，$D_{TD(i)}$ 为第 $i$ 个 *TD* 所包含的数据字节数，$P_{TD(i)}$ 表示第 $i$ 个 *TD* 传输的协议消耗字节数。USB 数据帧结构如图 2 所示，每帧包含一个 Frame Pointer 和若干 *TD*、*QH*（Queue Heads），其中一个 *QH* 指向的所有 *TD* 对应同一个 USB 设备[4-5]。

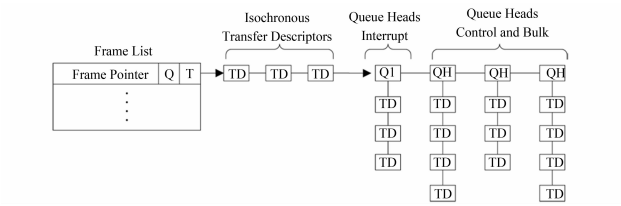



图 2 帧列表结构

Fig. 2 Frame list structure

上述事务调度机制由于不具备学习能力，块传输时数据帧的形成受事务属性的影响，使 Frame List 中某些数据帧对 USB 带宽资源的利用不够充分，造成资源的浪费。因此，针对上述问题，本文结合强化学习算法对 USB 传输事务进行智能调度，使得带宽资源利用更充分。

# 2 基于强化学习的事务调度

## 2.1 强化学习

强化学习基本模型如图 3 所示。在学习过程中，agent 选择一个动作 $a$ 作用于环境，环境接受该动作后转移至新状态 $s$，同时产生一个强化信号 $r$ 反馈给 agent，agent 再根据强化信号和环境当前状态选择下一个动作，选择原则是使受到正奖赏的概率增大[6]。

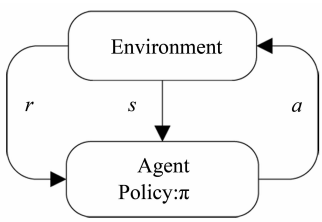



图 3 强化学习模型

Fig. 3 Reinforcement learning model

强化学习算法的目的是学习寻找一个策略 $\pi: S(A$，使得每个状态 $s$ 的值 $V^{\pi}(s)$（或）$Q^{\pi}(s, a)$ 达到最大[6]。

$$V^{\pi}(s) = E_{\pi}\{\sum_{i=0}^{\infty} \gamma^{i} r_{i+1} \mid s_0 = s\}, 0 \leqslant \gamma < 1 \tag{2}$$

$$Q^{\pi}(s,a) = E_{\pi}\{r(s,a) + \gamma V^{\pi}(\delta(s,a)) \mid s_0 = s, a_0 = a\} \tag{3}$$

$$V^{*}(s) = \max_{\pi}(V^{\pi}(s)) \tag{4}$$

$$Q^{*}(s,a) = \max_{\pi}(Q^{\pi}(s,a)) \tag{5}$$

其中，$r_i$ 表示在 $i$ 时刻的立即回报，$\gamma$ 为折扣

因子，$\delta(s, a)$ 表示在状态 $s$ 时执行动作 $a$ 的结果状态，$V^*(s)$ 和 $Q^*(s, a)$ 为最优值函数，其相应的最优策略为

$$\pi^*(s) = \text{argmax}\, V^*(s) \text{ 或}$$
$$\pi^*(s) = \text{argmax}\, Q^*(s,a) \quad (6)$$

SARSA 算法是强化学习中的一种在策略（on-policy）的时序差分算法[7]，它由 Rummery 和 Naraajan 提出，Singh 证明了在策略算法的收敛性。SARSA 在进行迭代更新时用到五元组（$s$, $a$, $r$, $s'$, $a'$），其中 $s$ 表示当前状态；$a$ 表示当前状态下选择的动作；$r$ 是动作 $a$ 的奖励；$s'$和 $a'$则分别是后续的状态和动作。SARSA 算法中行为值函数采用实际的 $Q$ 值进行迭代，通过式子（7）迭代规则来获得 $Q^*(s, a)$：

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \quad (7)$$

式中 $Q(s_t, a_t)$ 代表 $t$ 时刻的 $Q$ 值；$s_t$ 为状态；$a_t$ 为当前所采取的动作；$r_t$ 为奖励函数值；$\gamma$ 为折扣因子，表示未来回报相对当前回报的重要性；$\alpha$ 为学习率，其中 $0 \leqslant \alpha \leqslant 1$ 和 $0 < \gamma \leqslant 1$。

### 2.2　基于 Sarsa 学习算法的调度模型

1）状态集合

本文中将块传输事务按照传输数据包的大小进行分类，每一类传输事务用 $Trans_i$，$i = (1, 2, \cdots, n)$ 表示。将每种传输事务在每一帧中所分配的数量组合定义为 USB 总线传输状态：$S = \{Num_{Tran1}, Num_{Tran2}, Num_{Tran3}, \cdots, Num_{TranN}\}$。

2）动作集合

事务调度过程中，对每种 *TD* 的操作将影响 USB 总线上的传输状态，因此将对每种 *TD* 的操作定义为动作集合，$A_i = \{Add_{Tran(i)}, Replace_{Tran(i)}, Null\}$，其中 $Add_{Tran(i)}$ 表示增加第 $i$ 个传输事务，$Replace_{Tran(i)}$ 表示替换第 $i$ 个传输事务，*Null* 表示不采取任何动作。

3）动作探索策略

Sarsa 学习通过探索—利用过程发现最优目标，“利用”过程选择最高 $Q$ 值所对应的动作；“探索”过程则是随机选择动作以弥补“利用”的不足[7]。本文采用“$\varepsilon$—贪婪探索策略”，在状态 $s$ 下以小概率 $\varepsilon$ 随机选择动作 action，以 $1-\varepsilon$ 选择具有最大 $Q$ 值的动作。

4）立即回报

将 USB 总线在状态 $s_i$ 下执行动作 $a_i$ 转移到状态 $s_j$ 后，总线带宽利用率的增加值定义为立即回报，$r(s_i, a_i) = Utilize\_Increase_{ij}$。

通过对 USB 总线传输状态 $S$ 进行感知，并根据策略 $\pi$ 选取动作对块传输事务 $Trans_i$ 进行操作，观察回报 $r$ 与状态 $s$ 不断的改进策略 $\pi$，使得 USB 带宽有效利用率提高，基于 SARSA 学习算法的调度方法描述如下所示。

1）建立状态空间 *State* 和动作空间 *Action*，初始化 $Q$ 值表和 $\varepsilon$；

2）初始化 USB 总线传输状态 $s$；

3）映射出当前状态 $state_i$，并建立一个符合当前状态的数据帧；

4）根据“$\varepsilon$—贪婪探索策略”选择动作 $action_i$；

5）根据所选动作对该帧进行传输事务的增加或替换，构建新的数据帧，观察回报和状态；

6）更新 $Q$ 值表；

7）更新状态与动作；

8）若 USB 带宽有效利用率未达到饱和，则跳到步骤 3，否则结束。

## 3　仿真与分析

### 3.1　仿真建立

为验本文证算法在本应用中的有效性，将本文设计的调度方法与系统本身采用的“先到先得，尽力而为”的调度方法按照以下描述场景进行仿真实验对比。

本文针对 USB Full-Speed（12Mbps）传输速率中的 3 类块传输事务进行建模仿真。该 3 类块传输事务分别为：*Trans*（1）－16Bytes、*Trans*（2）－32Bytes、*Trans*（3）－64Bytes。上述三种传输事务在全速块传输的每一帧中所能传输的最大次数分别为 51、33、19[1]。因此，USB 总线状态集合 $S = \{Num_{Tran1}, Num_{Tran2}, Num_{Tran3}\}$，$0 \leqslant Num_{Trans1} \leqslant 51$，$0 \leqslant Num_{Trans2} \leqslant 33$，$0 \leqslant Num_{Trans3} \leqslant 19$。贪婪探索策略中随机选择动作的概率 $\varepsilon$ 初值设为 0.01，每次迭代后将其乘以 0.99 以降低随机选择动作的概率。

### 3.2　实验结果与分析

针对文献［1，5］所描述的系统自身事务调度方法和本文设计的方法，分别进行独立仿真实验，并对 USB 带宽有效利用率、吞吐量用、有效数据传输量、带宽剩余量等指标进行分析。两种方法仿真结果如图 4 所示，图中 Sarsa 曲线代表本文方法，System 代表系统调度方法，横坐标每一个值代表 10 次仿真，纵坐标每一个值为每 10 次仿真的平均值。本实验结果对应 SARSA 算法迭代算子中的折扣因子 γ、学习率 α 分别取 0.9 和 0.5。经多

次仿真实验，当折扣因子 γ 取值较小（0.5、0.8）时，容易收敛于次优策略，仍造成一定带宽资源浪费，且收敛较慢；取值 0.9 及以上时对实验结果影响不明显，且能获得较好的效果。学习率 α 的取值较小（0.1、0.2）或越接近 1 时，容易产生扰动，影响收敛速度，造成带宽利用率波动。

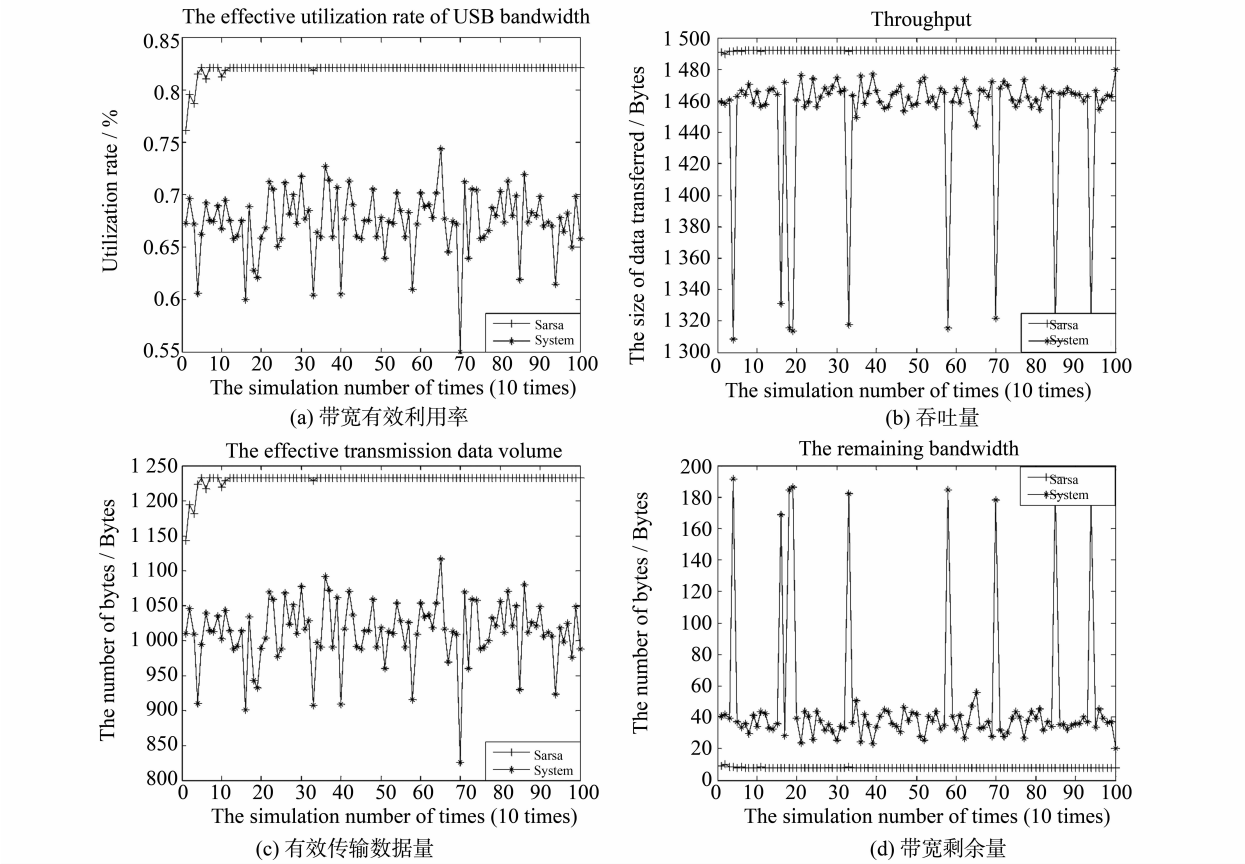



图 4 仿真结果

Fig. 4 The simulation result

从图 4 中可知，USB 软件系统本身的调度方法不具备学习能力，在整个过程中上述四种指标均处于波动状态。基于 SRASA 算法的调度方法在学习初期，由于系统未获得环境的全部先验知识，带宽利用率较低，但随着学习次数的增加，经过约 120 次左右的学习后各项指标均达到一个较好值，且保持稳定。

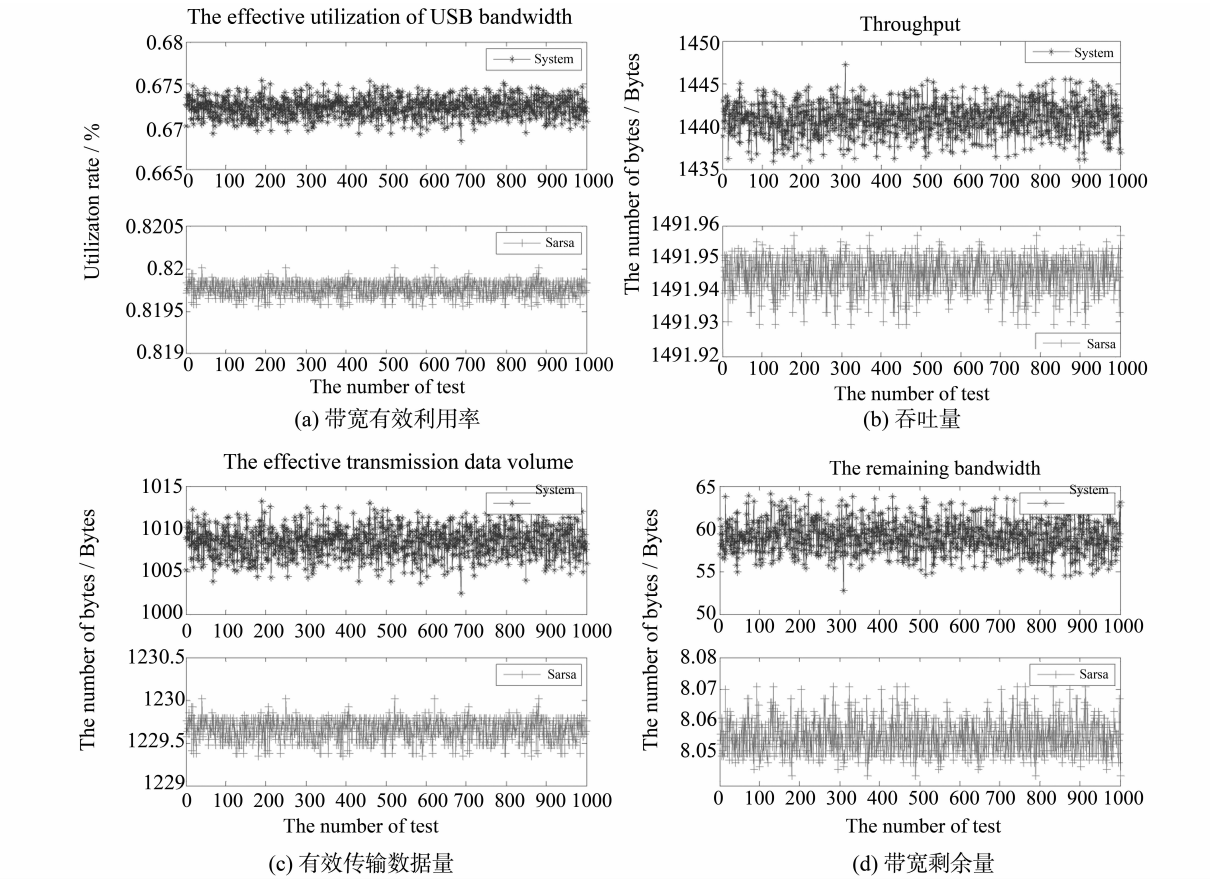
为进一步验证算法的性能，分别对上述两种方法进行 1 000 次实验测试，实验结果如图 5 所示，两种方法各项指标对比如表 1。

从表 1 两种方法 1 000 次实验测试结果对比可看出，基于 SARSA 算法的调度方法的各项指标均优于系统方法，且波动较小，性能稳定。前者相对后者平均有效利用率提高 21.94%，平均吞吐量提高 3.53%，平均有效数据传输量提高了 21.94%，平均剩余带宽降低了 86.36%。可看出采用 SARSA 学习算法后的 USB 块事务调度能够使 USB 带宽资源得到充分的利用，提高数据传输的效率。

表 1 1 000 次测试实验结果统计分析

Table 1 Statistical analysis of 1 000 testing the result of the experiment

| | 均值 | | 方差 | |
|---|---|---|---|---|
| | 系统方法 | 本文方法 | 系统方法 | 本文方法 |
| 有效利用率 | 67.23% | 81.98% | 3.394 7 | 2.688 6e－5 |
| 吞吐量 | 1 441.0 | 1 491.9 | 3.394 7 | 2.688 6e－5 |
| 有效传输数据量 | 1 008.4 | 1 229.7 | 2.826 7 | 0.015 7 |
| 带宽剩余量 | 59.038 1 | 8.055 0 | 1.256 3e－6 | 6.960 0e－9 |




(a) 带宽有效利用率　(b) 吞吐量　(c) 有效传输数据量　(d) 带宽剩余量

图 5　1 000 次测试实验结果

Fig. 5　The result of 1 000 tests

### 3.3　收敛性分析

SARSA 学习算法是一种在线策略强化学习算法，动作探索策略的选择对算法的收敛性具有关键作用。文献［8］给出了 SARSA 学习算法的收敛性定理，并在分别采取 GLIE 和 RRR 渐进贪心动作探索策略的情况下，对该定理进行了证明。

**定理 1**[7]　对一个有限“动作—状态”空间的 MDP，将一个固定的 GLIE 学习策略 $\pi$ 作为概率 $\Pr(a \mid s, t, n_t(s), Q)$ 的一个集合。假设在时间步 $t$ 时，根据策略 $\pi$ 选择动作 $a_t$，此时 $Q = Q_t$，$Q_t$ 的值根据式（7）进行计算。当同时满足以下三个条件时，$Q_t$ 收敛于最优值 $Q^*$，策略 $\pi_t$ 收敛于最优策略 $\pi^*$：

1）$Q$ 值存储于一个查找表中；

2）学习率 $\alpha$ 满足 $0 \leqslant \alpha_t(s, a) \leqslant 1$，$\sum_t \alpha_t(s, a) = 0$，$\sum_t \alpha_t^2(s, a) < \infty$，并且，当且仅当 $(s, a) = (s_t, a_t)$ 时 $\alpha_t(s, a) = 0$；

3）回报值方差 $\mathrm{Var}\{r(s, a)\} < \infty$。

本文选择三种块传输所有组合中满足带宽限制条件的 6 450 种组合作为状态集合，并采用查找表对 $Q$ 值进行存储。根据文献［8］的证明可知本文所采用的“$\varepsilon$—贪婪探索策略”属于 GLIE 学习策略，且满足渐近贪心和对“状态—动作”空间无限遍历的条件。

根据以上描述，本文方法中学习率 $\alpha_t(s, a) = 0.5$，因此 $t(\infty$ 时，$\sum_t \alpha_t(s, a) = \infty$。又因为 $0 < \alpha_t(s, a) < 1$，因此 $\alpha_t^2(s, a) < \alpha_t(s, a)$，所以 $\sum_t \alpha_t^2(s, a) < \sum_t \alpha_t(s, a)$，即 $\sum_t \alpha_t^2(s, a) < \infty$。

回报值方差定义如下：

$$\mathrm{Var}\{r(s,a)\} = \frac{1}{n} \times \sum_{i=1}^{n} \left( r_i(s,a) - \frac{1}{n} \times \sum_{j=1}^{n} r_j(s,a) \right)^2 \quad (8)$$

由于 $-1 < r(s_i, a_i) = Utilize\_Increase_{ij} < 1$ 可得，

$$0 \leqslant \left( r_i(s,a) - \frac{1}{n} \times \sum_{j=1}^{n} r_j(s,a) \right)^2 < 4 \quad (9)$$

将式（9）代入式（8），可得

$$\mathrm{Var}\{r(s,a)\} = \frac{1}{n} \times \sum_{i=1}^{n} \left( r_i(s,a) - \frac{1}{n} \times \sum_{j=1}^{n} r_j(s,a) \right)^2 < 4 < \infty \quad (10)$$

综上所述，本文中采用的基于SARSA算法的事务调度满足定理1的收敛条件，因此，该调度方法可通过一定次数的学习，获得最优调度策略 $\pi^*$。

## 4 结 论

带宽资源的有效利用对于大数据传输十分重要，本文针对包含多种块传输类型的情况，结合强化学习算法，设计了一种基于SARSA学习算法的传输事务调度方法。该方法在学习过程中，通过评估函数对所选动作进行评价、影响后续动作的选取，极大提高了传输事务的调度效率。本文通过建立模型并仿真，比较了系统调度方法和本文方法。仿真结果表明在相同传输环境下，本文所设计的基于SARSA算法的调度方法能够获取较高的带宽有效利用率，且提高USB总线吞吐量。

**参考文献：**

[1] COMPAQ, HEWLETT-PACKARD, INTEL. Universal serial bus specification[S]. Revision 2.0, 2000.

[2] HUANG C Y, KUO T W, PANG A C. QoS support for USB2.0 periodic and sporadic device requests [C]// IEEE International Real-Time Systems Symposium, 2004:395 – 404.

[3] INTEL. Universal host controller interface (UHCI) design guide[S]. Revision 1.1, Intel, 1996.

[4] MISSIMER E, LI Y, WEST R. Real-time USB communication in the quest operating system [C]//IEEE Real-Time Technology and Applications-Proceedings, 2013:11 – 20.

[5] ANDERSON DON, DZATKO DAVE, MINDSHARE INC. Universal serial bus system architecture [M]. 2nd ed. Addison-Wesley Educational Publishers Inc, 2001: 141 – 154.

[6] 舒波,李大铭,赵新良,等. 基于强化学习算法的公交信号优先策略[J]. 东北大学学报:自然科学版, 2012, 33(10):1513 – 1516.

[7] JIANG L B, JIANG H, WU S, et al. Decentralized cognitive MAC protocol based on SARSA [C]//IEEE International Conference on Communication Technology Proceedings, ICCT, 2012: 392 – 396.

[8] SINGH S, JAAKKOLA T, SZEPESVÁRI C, et al. Convergence results for single-step on-policy reinforcement learning algorithms [J]. Machine Learning, 2000, 38 (03): 287 – 308.