CMDRL: A Markovian Distributed Rate Limiting Algorithm in Cloud Networks

Lilong Chen[†]
chenlilong.cll@zju.edu.cn

Xiaochong Jiang[†]
xiaochong.jxc@zju.edu.cn

Xiang Hu[†]
xianghu@zju.edu.cn

Tianyu Xu[†]
ep_@zju.edu.cn

Ye Yang[§]
huaizhou.yy@alibaba-inc.com

Xing Li[†§]
lixing.lix@aliyun-inc.com

Bingqian Lu[†]
BingqianLu@zju.edu.cn

Chengkun Wei[†□]
weichengkun@zju.edu.cn

Wenzhi Chen[†□]
chenwz@zju.edu.cn

[†]Zhejiang University [§]Alibaba Cloud

ABSTRACT

As cloud networks continue to evolve, network traffic has experienced an exponential increase. The network architecture is progressively adopting a distributed structure to address this challenge. This architecture extensively utilizes technologies like gateway clusters and Equal-Cost Multi-Path (ECMP) routing, enabling traffic from individual tenants to be routed through multiple pathways. As a result, distributed rate limiting (DRL) has emerged as an essential aspect. Nonetheless, the shift from centralized to DRL has encountered obstacles, with the associated algorithms grappling with simplicity, precision, and applicability issues. Consequently, our research seeks to reconceptualize the issue of DRL from a theoretical standpoint to discover a more holistic and efficacious solution.

In this paper, we introduce CMDRL, a novel Markovian DRL algorithm that conceptualizes the problem as a random walk on a graph and frames it within a Markov model premised on two widely accepted assumptions. We establish that the converging model exhibits a bounded mixing time, serving as a more comprehensive metric for assessing convergence velocity. The algorithm's benefits include simplified input requisites, greater versatility across diverse contexts, and improved accuracy. Our evaluation results show that the algorithm attains a favorable convergence, evidenced by an L1-norm error of 0.063 after 50 iterations, demonstrates a rapid adaptive response with a 72.9% decrease in error within 20 iterations, and achieves a swift mixing time, requiring only 33 iterations to reach an error threshold of 0.05. Additionally, We evaluate the effectiveness of various optimization strategies for Markov chains in addressing the DRL problem.

□Co-corresponding authors

APNet 2024, August 3–4, 2024, Sydney, Australia

ACM ISBN 979-8-4007-1758-1/24/08…$15.00
https://doi.org/10.1145/3663408.3663417

CCS CONCEPTS

• **Networks** → **Control path algorithms**; **Data center networks**;

KEYWORDS

Distributed Rate Limiting, Cloud Networking, Markov Model

ACM Reference Format:
Lilong Chen, Xiaochong Jiang, Xiang Hu, Tianyu Xu, Ye Yang, Xing Li, Bingqian Lu, Chengkun Wei, Wenzhi Chen. 2024. CMDRL: A Markovian Distributed Rate Limiting Algorithm in Cloud Networks. In *The 8th Asia-Pacific Workshop on Networking (APNet 2024), August 3–4, 2024, Sydney, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3663408.3663417

1 INTRODUCTION

Traffic rate limiting in cloud data centers is crucial for cloud service providers (CSPs) to achieve predictability, performance isolation, elasticity, and high availability. By enforcing a multi-level (from per-flow to per-tenant) rate limiting policy on each Network Function (NF) node, CSPs utilize network capacity efficiently by preventing resource contention, maintaining service level agreements (SLAs) and provide a consistent user experience. Typically, rate limiting relies on stateful tables for bookkeeping, followed by token buckets or other traffic shaping mechanisms to regulate traffic flow.

With the trends of high link rate, multi-path transport protocol, and network redundancy, CSPs begin to deploy NFs (e.g., firewall, IDS) in a clustered manner. This makes implementing collective rate limiting across distributed nodes for per-tenant traffic imperative. While CSPs exhibit a strong demand for DRL to guarantee per-tenant bandwidth envelope, as depicted in Figure 1, the spraying traffic across distributed nodes adds complexity. The complexity for an ideal DRL algorithm can be summarized in three aspects: First, the DRL should be *precise* to ensure that the aggregate rate converges to the global rate limit. Second, the nodes in the cluster should be *well-coordinated*, meaning that they need to swiftly synchronize information and converge to the target local limit rate as traffic distribution changes. Finally, the DRL must be *versatile*

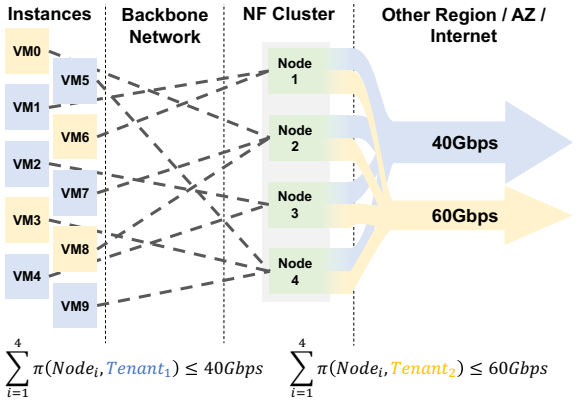

Figure 1: A typical scenario of DRL in distributed cloud gateways. The π indicates the specific rate limit of each gateway node, and the aggregate rate for each tenant is restricted to the maximum bandwidth envelope. The VMs belonging to different tenants are respectively annotated with blue and yellow.

enough to accommodate various traffic patterns driven by different congestion control algorithms.

Previous works, however, fail to meet the performance requirements of DRL. In the ensuing discussion, we identify and elaborate on three critical challenges that arise within current cloud environments.

Challenge 1. Dependence on inaccurate or complex estimators to adjust the distributed nodes' limit rate. Several studies [22] have enabled distributed nodes to approximate global parameters, effectively emulating the capabilities of a centralized rate limiter. For instance, the Global Random Drop (GRD) algorithm [22] permits each node to independently deduce the collective rate limit as a basis for calculating the packet loss probability. Nonetheless, such estimations are inherently approximate and suffer from temporal delays. Comparable methodologies manifest in a variety of algorithms that employ techniques such as sliding windows [10], leaky buckets [13], and token buckets [26]. Additionally, the Cloud Control with Constant Probabilities (C3P) algorithm [25] relies on TCP's square-root formula to gauge packet loss rates for rate limit adjustments, a method that is not only imprecise but also adds to the system's complexity.

Challenge 2. Inaccuracy and instability of the global limit rate and the nodes' limit rate. The GRD algorithm eschews explicit rate-limiting parameters, instead utilizing the forwarding rates as a proxy for limit rates. Furthermore, numerous DRL algorithms that depend on coarse estimations are inherently incapable of maintaining precise global limit rates. An additional limitation is the susceptibility of algorithms such as C3P and Distributed Deficit Round Robin (D2R2) [25] to transient oscillations in global limit rates in response to abrupt traffic shifts, requiring time to regain stability. Moreover, both C3P and D2R2 lack the provision for manual configuration of limit rates at individual nodes.

Challenge 3. Lack of universality, whose correctness depends on specific mechanisms in particular scenarios. TCP continues to be the principal transport protocol for Internet traffic, with numerous DRL algorithms tailored for TCP connections. For example, the C3P and the Flow Proportional Share (FPS) algorithms leverage TCP's native congestion control mechanisms. Alternatively, some approaches, such as DCTCP [1], modify the congestion control algorithm within the TCP protocol to regulate transmission rates, utilizing multi-bit feedback from consecutive ECN marks to achieve their objectives. Nevertheless, these methods do not provide a universally applicable solution across diverse network scenarios.

In this paper, we propose CMDRL, a new DRL algorithm based on Markov chains. We employ a simpler *input rate* in place of complex estimations and the Markovian convergence process to achieve high accuracy. We establish the convergence model and delineate an upper bound for the convergence speed in prevalent topologies under a constant global limit rate. To evaluate the convergence speed, we utilize the Mixing Time as a universal metric. The input and output of our model are both straightforward and versatile, independent of any specific network protocol.

Our preliminary simulation results demonstrate the efficacy of these designs. Given a predetermined limiting distribution, the algorithm requires 50 iterations to reduce the L1-Norm error to 0.063. Following a sudden change in the limiting distribution, the L1-Norm error decreased by 72.9% after 20 iterations. We introduce mixing time as a more comprehensive criterion for convergence speed, and corroborate the theoretical upper bound of 50 rounds. Subsequently, we transfer the optimization of Markov chains to this problem, improving the mixing time.

2 BACKGROUND AND MOTIVATION

2.1 Problem Statement

In a centralized rate limiting system with a threshold a, an input rate x, and an output rate y, the system's transfer function is described by $y = \max\{a, x\}$. Conversely, in distributed environments, x, y and a each consist of contributions from n distinct nodes, with each node adhering to the relationship $y_i = \max\{a_i, x_i\}$. The limit rate for each node is variable and time-dependent, represented by the vector $\mathbf{a} = (a_1, ..., a_n)^T$. The vector $\pi = (\pi_1, ..., \pi_n)$ denotes the composite output rates. Collectively, these n nodes function analogously to a single centralized rate limiter, possessing an aggregate threshold limit, with the limit rate $a = \pi_1 + ... + \pi_n$.

A DRL algorithm is designed with three primary objectives:
1. π may not be directly computed, and $\mathbf{a}$ may change over time, but $\mathbf{a} \to \pi^T$ after a sufficiently long time.
2. The algorithm aims to achieve the fastest convergence speed for any given input.
3. π satisfies predefined fairness assumptions; for example, each node has an equal packet loss rate.

Assumption 2.1 (Linear Transform). *Let $\boldsymbol{a}_{i+1}$ be the limit values at the next point. Then, $\boldsymbol{a}_{i+1}$ is a linear transformation $\boldsymbol{a}_i$, i.e., there exists a matrix $A \in \mathbb{R}^{n \times n}$ such that $\boldsymbol{a}_{i+1} = A\boldsymbol{a}_i$.*

The exclusion of historical data and linear dependencies dramatically simplifies the design of the algorithm. Notably, this premise is upheld by both the C3P and D2R2 algorithms. The assumption constitutes a constraint on the algorithm rather than the actual scenario.

Assumption 2.2 (Stable Input Rate). *The matrix* $(s_1, ..., s_n)$ *varies with time; however, it will remain stable within the mixing time (see Section 3.3).*

This assumption is essential for all algorithms that rely on convergence, even though most such algorithms do not explicitly state it. The assumption seems to limit the input rate during actual deployment; however, in our implementation, we discover that it is easily satisfied with shortened mixing times (see Section 4).

2.2 Markov Chains

Let $X(t), t \geq 0$ be a stochastic process. Given the moments $t_1 < t_2 < ...$, if the conditional distribution function satisfies

$$\begin{aligned} &P\{X(t_n) \leq x_n | X(t_{n-1}) = x_{n-1}, ..., X(t_1) = x_1\} \\ &= P\{X(t_n) \leq x_n | X(t_{n-1}) = x_{n-1}\} \end{aligned} \tag{1}$$

then $X(t)$ is called a Markov process, and $\{X(t_n)\}_{n\geq 1}$ constitutes a Markov chain. Denote the transition probabilities as $P\{X(t_n) = i | X(t_m) = j\} = P_{ij}(n, m)$. We only consider stationary Markov processes, that is, those that satisfy $P_{ij}(n, m) = P_{ij}(n - m)$, with a one-step transition matrix being the stochastic matrix $P = (P_{ij}(1))_{n\times n}$.

We have discovered a connection between the Markov property and the DRL problem:

- The Markov process's probabilistic framework mandates that the sum of probabilities across all states is one at any moment. By interpreting the probability associated with each state as the normalized limit rate for each node, the Markov property inherently maintains the total limit rate as constant.
- Equation 1 posits that the future state of a Markov chain is exclusively determined by its present state. Additionally, the Chapman-Kolmogorov equations in matrix form indicate that the state at time n can be derived from a linear transformation applied to the initial state, which is consistent with Assumption 2.1.
- Markov chains may have a limiting distribution. Given the prerequisites of Assumption 2.2, it is possible to design appropriate chains that guarantee the convergence of normalized limit rates.

We can use a straightforward metric instead of the packet loss rate to adapt the C3P algorithm for Markovian processes. Assuming that the input rate $s > a$, we replace the packet loss rate with $\frac{s-a}{s} = 1 - \frac{a}{s}$, and yield an adjustment formula that is solely dependent on the input rate and the limit rate:

$$a_k \leftarrow a_k + \eta \sum_{(i,k)\in E} \left(\frac{a_i}{s_i} - \frac{a_k}{s_k} \right) \tag{2}$$

Assuming full adjacency among nodes and denoting the i^{th} adjustment round with subscript i, the update equation for the subsequent round is given by $\mathbf{a}_{i+1} = P^T \mathbf{a}_i$, where

$$P^T = \begin{pmatrix} 1 - \frac{\eta}{s_1}(n-1) & \frac{\eta}{s_2} & ... & \frac{\eta}{s_n} \\ \frac{\eta}{s_1} & 1 - \frac{\eta}{s_2}(n-1) & ... & \frac{\eta}{s_n} \\ ... & ... & & \\ \frac{\eta}{s_1} & \frac{\eta}{s_2} & ... & 1 - \frac{\eta}{s_n}(n-1) \end{pmatrix} \tag{3}$$

Let $\eta \leq \frac{\min_k\{s_k\}}{n-1}$, ensuring P is a stochastic matrix, with P_{ij} representing a one-step transition probability. Typically, P exhibits symmetrically distributed zero entries in its off-diagonal positions, yet it retains its stochastic nature. We aim to use Markov chains to model the DRL problem and to address three issues in Section 2.1.

3 DESIGN OVERVIEW

3.1 Markov Chain Modeling

Considering an undirected graph $G = (V, E)$ that depicts the topology of distributed nodes, we recast the DRL problem into a lazy random walk on G. An edge $(i, j) \in E$ exists if and only if there is a nonzero probability for node i to transition to node j, and conversely, for node j to transition to node i. Representing the transition matrix by P, we establish that both $P_{ij} > 0$ and $P_{ji} > 0$, although P_{ij} is not obliged to equal P_{ji}. The random walk is deemed lazy, signified by $P_{ii} > 0$. The matrix P characterizes a finite Markov chain with the state space denoted by $\mathcal{X}$.

Let $\mathbf{a}_i$ represents the rate vector for all nodes at time i, and the DRL algorithm updates the vector as $\mathbf{a}_{i+1} = P^T \mathbf{a}_i$. Considering node k,

$$a_k \leftarrow P_{kk} a_k + \sum_{(i,k)\in E} P^T_{ki} a_i \tag{4}$$

Subsequently, we model this random walk problem as Markov chains. Next, by scrutinizing the two principal characteristics of Markov chains, we reframe the three issues delineated in Section 2.1 into three corresponding problems.

3.1.1 Stationary Distribution and Limit Distribution. When the input rate is stable, the final rate vector becomes $a = (P^T)^\infty a_1$, corresponding to the limiting distribution of the Markov chain. This prompts the query: *Does the chain have a limiting distribution? (Section 3.2)*

In practical rate-limiting services, the limiting distribution should be predetermined. Determining an appropriate limiting distribution necessitates considering numerous factors, such as traffic characteristics, node performance, and fairness among competing flows. Therefore, the next question is *How to determine the limiting distribution? (Section 3.4)*

3.1.2 Convergence Speed. Despite the convergence guarantee under Assumption 2.2, it is imprudent to presume an indefinitely prolonged stabilization period. Therefore, it is imperative to facilitate the swift convergence of limit rates to the limiting distribution. This consideration leads us to convergence speed: *How to obtain the fastest convergence speed? (Section 3.3)*

The following three subsections will address these three problems.

3.2 Existence of Limit Distribution

3.2.1 Topologies with Limit Distribution. A limiting distribution may be absent if the graph G yields reducible transition matrices. We explore lazy random walks on various topologies with a limiting distribution.

Complete Graph. The transition matrices of complete graphs are elementwise positive, meeting the requirements of Lemma A.1. Consequently, a limiting distribution is ensured to exist.

Complete Bipartite Graph. Let $V = X \cup Y$, while each vertex in X is connected to each vertex in Y by a single edge, then G is a complete bipartite graph. The associated transition matrix P fulfills the prerequisites of Lemma A.2, resulting in P^2 being elementwise positive. By invoking Lemma A.1, we can confirm the existence of a limiting distribution.

Cycle. Lemma A.3 states that a limiting distribution exists for an irreducible and aperiodic chain. In a cycle, each vertex can transition to its neighboring vertices, implying that for any two states $x, y \in \mathcal{X}$, there is a positive integer t such that $P^t(x, y) > 0$, making P irreducible. Moreover, in a lazy random walk, $P(x, x) > 0$ for any state x ensures that P is aperiodic. Given these conditions, P adheres to the criteria of Lemma A.3, confirming the existence of a limiting distribution.

3.2.2 Construction of Transition Matrix. A multitude of methods have been proposed to deduce the transition matrix P from a prescribed stationary distribution π [2, 6, 9, 11, 15, 20]. Among these, the Metropolis-Hastings (MH) algorithm [11] is deemed the most appropriate for DRL in cloud networks (see Appendix B). The MH algorithm is advantageous as it necessitates only an initial stochastic matrix and a desired stationary distribution for inputs. In contrast, other algorithms that offer higher optimization typically require lifting of the state space [4] or the introduction of extra parameters, such as breaking detailed balance [17], applying coloring techniques [27], implementing perturbations [24], or employing other strategies. Moreover, the MH algorithm supports distributed computing of the transition matrix, facilitating the distribution of the controller's computational burden across network nodes. Notably, with equivalent computational complexity, the MH algorithm can accommodate asymmetric transitions, unlike the traditional Metropolis algorithm [20].

3.3 Convergence Speed and Mixing Time

We measure the convergence speed using the Total Variation (TV) Distance and the Mixing Time. We define the TV Distance between the distribution after several iterations and the limiting distribution as $d(t) = \max_{x \in \mathcal{X}} \|P^t(x, \cdot) - \pi\|_{TV}$ and define the Mixing Time as

$$t_{min}(\epsilon) = \min\{t : d(t) \leq \epsilon\} \tag{5}$$

where ϵ is a hyperparameter that is business-specific and may influence the assessment of algorithmic approaches.

In certain instances, it is possible to establish a theoretical upper bound for the Mixing Time. For instance, an upper bound about the cycle topology is delineated in Appendix C.

For general graphs, well-established methodologies exist to deduce mathematically both lower bounds [8, 12, 21] and upper bounds [7, 16, 19] on similar convergence metrics. Lemma A.5 offers an upper bound on the L^2-Norm for symmetric stochastic matrices based on the second largest eigenvalues, λ^*. Furthermore, the Rayleigh-Ritz variational principle [14] provides bounds for λ^*. In scenarios involving asymmetric matrices, eigenvalue theory [7] furnishes an upper bound for the mixing time $t_{mix}(\epsilon)$ pertinent to the context of lazy random walks.

While λ^* indicates the upper bound of convergence speed, as discussed in Section 5.2.1, $t_{mix}(\epsilon)$ serves as a more holistic measure, and the selected value of ϵ significantly influences the evaluation of convergence speed.

3.4 Fairness Assumptions

We generate the limiting distribution π based on fairness assumptions. Let $\theta_1, ..., \theta_m$ be environmental parameters (e.g., node performance, user payments, etc.), the fairness assumption is a mapping from $(s_1, ..., s_n; \theta_1, ..., \theta_m)$ to π. A trivial assumption is that each node's limit value is the same, which sets π_i to $\frac{1}{n}$. We propose a slightly more complex and practically viable fairness assumption, Flow Proportionate Fairness (FPF), which means π_i is proportional to s_i. We set

$$\pi_i = \frac{s_i}{s_1 + ... + s_n} \tag{6}$$

Specifically, if s denotes the traffic of a flow and we use $\frac{s-a}{s}$ to describe the packet loss rate at the receiver side, then as $a \rightarrow \pi$, it holds that $\frac{s_i - a_i}{s_i} = \frac{s_j - a_j}{s_j}$, which implies that this assumption represents per-flow fairness in packet loss. Similarly, if s represents the total traffic volume at the granularity of tenants, this assumption signifies that all user traffic is rate-limited in the same proportion.

We employ the MH algorithm to calculate the transition matrix. The algorithm running on node k requires only the information of the ratio $\frac{\pi_k}{\pi_{neigh}}$, that is, the ratio of the limiting distributions of the node and its neighbors. The flow proportionate fairness assumption enables us to readily obtain this information, as $\frac{\pi_k}{\pi_{neigh}} = \frac{s_k}{s_{neigh}}$, where s_k and s_{neigh} can be directly measured and only necessitate a single exchange of information between neighbors.

4 IMPLEMENTATION

Our model is designed to be universally applicable, not contingent upon assumptions tailored to specific use cases, thus making it suitable for a broad range of DRL systems, including TCP, UDP, QPS, RDMA, and others.

Without loss of generality, We conduct simulation with 10 nodes arranged in a cycle to avoid excessive communication overhead. As depicted by the orange segment in Figure 2, the Controller generates the initial $\hat{P}$ and the limiting distribution π, distributing them to the nodes across the network. If employing the MH algorithm, the subsequent adjustment of the limit rate by node i becomes dependent on its adjacent nodes' parameters, namely $\hat{P}_{ij}, \hat{P}_{ji}, \pi_i$ and π_j in Equation 15. As a result, it is only necessary for the Controller to disseminate π to the nodes, and the nodes can then independently execute the distributed MH algorithm. The blue section of the figure illustrates the iterative process wherein neighboring nodes exchange information and update their limit rates in each iteration. Essential to this exchange is the variable a_i, whereas the inclusion of s_i is discretionary, used primarily to signal the need for updating π if s_i experiences significant alterations, contravening Assumption 2.2. Subsequently, at the commencement of the following round, all nodes update their limit rates.

The task of generating π can be carried out by the Controller or the distributed nodes. Given that the Controller has access to a view of the input rates for all nodes, it can determine when an update of π is necessary. For the latter case, by adopting a more stringent constraint predicated on the fairness assumption, expressed as

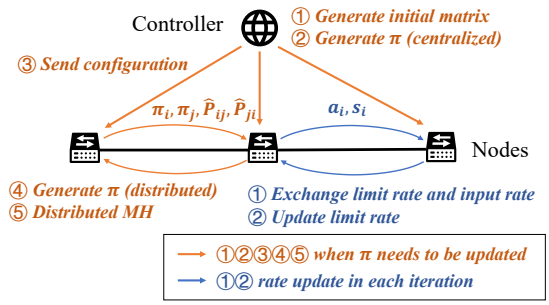

Figure 2: DRL system designed for the cycle topology.

$\frac{\pi_i}{\pi_{i+1}} = f(s_i, N(s_i))$ ($\frac{\pi_i}{\pi_{i+1}} = \frac{s_i}{s_{i+1}}$ for FPF) with $N(s_i)$ denoting the neighboring input rates, it is possible to eliminate the Controller at other times apart from the initial step of generating $\hat{P}$, allowing individual nodes to decide whether an update of π is required. The pseudocode for this algorithm is shown in Algorithm 1.

To simulate this process, we developed a program that deploys multiple processes to represent the Controller and the nodes. We assume that all nodes possess synchronized clocks and that the Controller dispatches configurations synchronously. While inter-node communication experiences latency, it is imperative that messages from adjacent nodes are received within the current iteration. The node k only retains the k^{th} column of the transition matrix P, which corresponds to the k^{th} row of P^T. The distributed nodes only need to perform floating-point multiplication, which manifests in the entire system as a Markov process. Implementing multiplication for doubles in software is quite straightforward.

The duration of each iteration is contingent upon the network configuration as well as the efficiency of the algorithm implementation. To illustrate, in our implementation, nodes communicate through RPC, with the RPC Request and Response sizes being 4B and 16B, respectively. In a typical data center network equipped with a bandwidth of 100G and a propagation delay of 1us, the transmission delay is measured in nanoseconds, dominated by the propagation delay. Considering concurrent requests to two neighbors by the RPC client, the total transmission-to-reception delay amounts to 2μs. If the algorithm is executed via software, the most substantial delay would stem from the computation of parameters and rate updates within the software, dictating the length of an iteration. Assuming a fixed duration of each round at 10ms, a Mixing Time of under 100 rounds would imply convergence achieved in less than 1 second. Given that the nodes involved in distributed rate limiting are responsible for handling tenants' business traffic, which generally exhibits minimal fluctuation within a one-second timeframe, this observation substantiates the validity of Assumption 2.2. It's important to note that the constancy of the global limit rate is maintained throughout that second; the "convergence time" referenced pertains only to the local rate limits. For most DRL applications, a local limit rate convergence time of 1 second would be considered satisfactory.

5 EVALUATION

In accordance with the design outlined in Section 4, we simulated a cycle topology network consisting of 10 gateway nodes. The controller sends configurations to these nodes synchronously, ensuring

Algorithm 1 Markovian Update

Controller

Input: $s = (s_1, ..., s_n)$, $\theta = (\theta_1, ..., \theta_m)$
1: $\hat{P}$ = random_init(n)
2: send_to_all_nodes($\hat{P}$)
3: **if** need_update **then** ▷ offload to nodes
4: π = fairness_assumption(s, θ)
5: send_to_all_nodes(π)
6: **end if**

Node

Input: $\hat{P}$
1: P_i = distributed_MH($\hat{P}$, $\frac{\pi_i}{\pi_{i+1}}$)
2: **while** next_iteration **do**
3: $a_{next} = P_{ii} a_i$
4: broadcast_to_neighbor(a, ...)
5: **while** msg = receive_msg() **do**
6: $a_{neighbor}$ = get_neighbor_rate(msg)
7: a_{next} += $P_{ik} a_{neighbor}$
8: **end while**
9: $a = a_{next}$
10: **end while**

reliable communication among adjacent nodes. This setup guarantees that every node receives messages from all its neighbors within a single iteration.

Within the scope of finite Markov chains, Lemma A.6 facilitates the evaluation of $\|\mu - \pi\|_1$, allowing for the analysis of the chain's dynamics.

We establish the threshold value ϵ for the Mixing Time at 0.05. In contrast to the necessity for high precision in the global total limit rate within DRL systems, a measure of error at the individual node level is permissible. Empirical evidence from random trials indicates that an ϵ of 0.05 yields an acceptably negligible error.

5.1 Convergence

Figure 3 depicts the convergence behavior of the algorithm for varying input rates. Initially, the input rates are initialized to $s_i = \frac{i}{55}$ $(1 \le i \le 10)$, and this configuration is retained for 100 iterations. In the subsequent phase, the input rates undergo a change such that $s_{2i-1} = s_{2i} = \frac{i+3}{60}$ $(1 \le i \le 5)$ and are then held constant. The fairness criterion follows Equation 6, starting each node with a rate of 0.1. We choose the odd-numbered nodes and the final node as representative samples in the figure.

The results depicted in Figure 3 demonstrate that the convergence rates for individual nodes invariably approach their theoretical limits. The L1-Norm error relative to the expected distribution decreases to 0.063 by the 50th iteration and further to 0.023 by the 100th iteration. Initial fluctuations in the limiting values are pronounced, suggesting a high sensitivity of the limit rates to the input rate modifications. After 20 iterations, the error diminishes to 0.161, which constitutes a 64.5% reduction from the initial iteration. Furthermore, Figure 4 illustrates the trend over 20 rounds following a change in the input rate, which results in the L1 error diminishing to 0.062, reflecting a 72.9% decrease, by the 120th round.

5.2 Convergence Speed

Different construction methods to create P from π resulting in varying mixing times. We utilize λ^* to distinguish between different

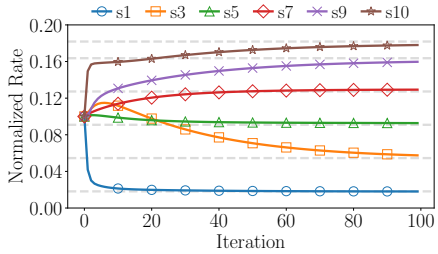

Figure 3: Convergence

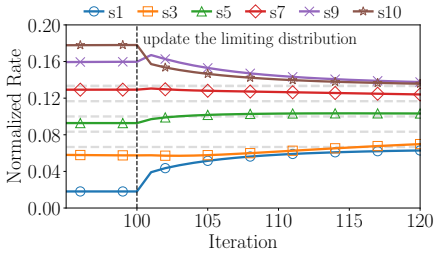

Figure 4: Traffic Variation

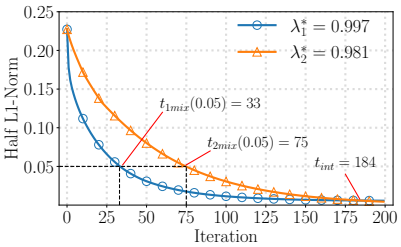

Figure 5: Mixing Time

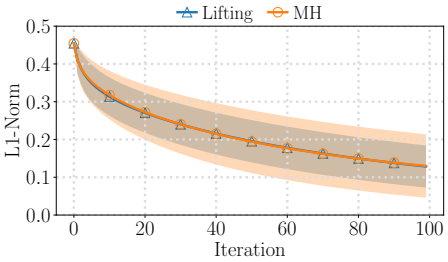

Figure 6: Lifting Optimation

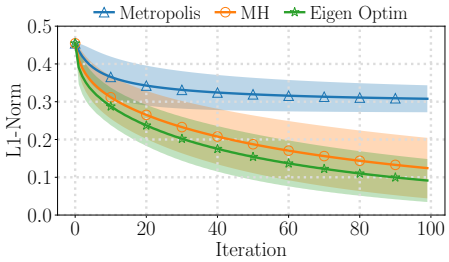

Figure 7: Eigenvalue Optim

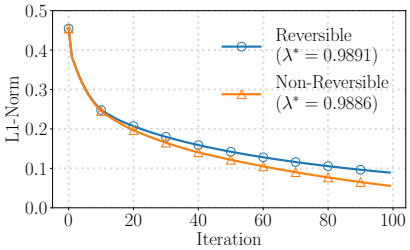

Figure 8: Non-Reversible Optim

matrices and demonstrate in Section 5.2.1 that smaller λ^* often leads to faster convergence rates (see 3.3).

Moreover, we evaluate the potential acceleration of convergence speed in DRL problems by optimizing the Markov chain in Section 8 to demonstrate that optimizing Markov chains is tantamount to optimizing DRL, with the former being widely studied [3–5, 17, 24].

5.2.1 Mixing Time and Eigenvalue. In this section, we evaluate the convergence speed of the experiments conducted in Section 5.1. We compute $\lambda_1^* = 0.997$ and $t_{1mix}(0.05) = 33$.

In multiple different random experiments, we find that although λ^* adequately defines the convergence behavior of the Markov Chain, considering only the long-term convergence in some cases is insufficient. We discover that short-term sensitivity is a critical metric. To illustrate this, we conduct experiments with another set of randomly selected initialization parameters, which resulted in a smaller $\lambda_2^* = 0.981$. In the first 50 iterations, for the second experiment could be worse than the first, with $t_{2mix}(0.05) = 75$. Indeed, after 184 iterations, the error in the second experiment begins to fall below the first, meaning that under the assumption of infinite time, the second experiment will achieve a faster convergence speed. The good news is we can control which scenario should be selected by choosing the ϵ parameter in the Mixing Time. In this case

$$t_{1mix}(\epsilon) > t_{2mix}(\epsilon) \quad \epsilon < 0.0118$$
$$t_{1mix}(\epsilon) \leq t_{2mix}(\epsilon) \quad \epsilon \geq 0.0118$$

In practical scenarios, when the precision requirement for the limit rates on each node is moderate (noting that the total rate is still constant while there is only some error in the rates allocated to the distributed nodes) and there is a need to reach the required precision more quickly, one may choose $\epsilon > 0.0118$. For instance, adopting $\epsilon = 0.05$ would make the first set of parameters more favorable.

5.2.2 Optimism of Markov Chains. The optimization of Markov chains can be categorized into two primary approaches. The first utilizes the Lifting method [4], which involves expanding the state space to enhance performance. The second approach modifies the MH algorithm, tailoring transition matrices to specific optimization strategies. This latter approach can be further partitioned into optimizations that focus on eigenvalue adjustments [3, 17] and those that employ non-reversible Markov chains [2, 18, 23, 24]. To compare these methods, we conduct randomized experiments and record the mean and range of one standard deviation for each algorithm's performance. We have chosen exemplar algorithms from each category for experimental evaluation. Figure 6 demonstrates the Lifting method's contribution [4] to reducing standard deviation. Figure 7 depicts the reduction in L1-Norm error achieved through eigenvalue-based optimizations [17] and contrasts the performance benefits of the MH algorithm [20] against the traditional Metropolis algorithm. Figure 8 highlights the superior convergence characteristics of non-reversible Markov chains [2] when compared under identical initial matrix conditions.

6 CONCLUSION

With the growing demand for cloud computing, the data center networks and the traffic characteristics are becoming increasingly complex, posing challenges to traditional DRL algorithms. We propose CMDRL, a new DRL algorithm based on Markov chains, which guarantees the accuracy, convergence and versatility through the Markov property. It is independent of network protocol and ensures algorithm performance by constraining the convergence rate upper bound. Leveraging extensive research on stochastic processes, this algorithm offers numerous optimization techniques. Through simulation experiments, we preliminarily confirm the effectiveness of the algorithm. We consider the actual system deployment as a future work. This work does not raise any ethical issues.

REFERENCES

[1] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. 2010. Data center TCP (DCTCP). In *Proceedings of the ACM SIGCOMM 2010 Conference (SIGCOMM '10)*. Association for Computing Machinery, New York, NY, USA, 63–74. https://doi.org/10.1145/1851182.1851192

[2] Joris Bierkens. 2014. Non-reversible Metropolis-Hastings. *Statistics and Computing* 26 (2014), 1213–1228. https://api.semanticscholar.org/CorpusID:38373899

[3] Joris Bierkens, Pierre Nyquist, and Mikola C. Schlottke. 2019. Large deviations for the empirical measure of the zig-zag process. *The Annals of Applied Probability* (2019). https://api.semanticscholar.org/CorpusID:209370547

[4] Fangu Chen, László Miklós Lovász, and Igor Pak. 1999. Lifting Markov chains to speed up mixing. In *Symposium on the Theory of Computing*. https://api.semanticscholar.org/CorpusID:7043817

[5] Persi Diaconis, Susan P. Holmes, and Radford M. Neal. 2000. ANALYSIS OF A NONREVERSIBLE MARKOV CHAIN SAMPLER. *Annals of Applied Probability* 10 (2000), 726–752. https://api.semanticscholar.org/CorpusID:6626758

[6] Persi Diaconis and Laurent Saloff-Coste. 1995. What do we know about the Metropolis algorithm? *J. Comput. Syst. Sci.* 57 (1995), 20–36. https://api.semanticscholar.org/CorpusID:7347257

[7] Persi Diaconis and Mehrdad Shahshahani. 1981. Generating a random permutation with random transpositions. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete* 57 (1981), 159–179. https://api.semanticscholar.org/CorpusID:120648396

[8] James Allen Fill. 1991. Eigenvalue bounds on convergence to stationarity for nonreversible markov chains. https://api.semanticscholar.org/CorpusID:118425802

[9] Bruce E. Hajek. 1988. Cooling Schedules for Optimal Annealing. *Math. Oper. Res.* 13 (1988), 311–329. https://api.semanticscholar.org/CorpusID:2287187

[10] Zijun Hang, Yang Shi, Mei Wen, and Chunyuan Zhang. 2019. TBSW: Time-Based Sliding Window Algorithm for Network Traffic Measurement. In *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 1305–1310. https://doi.org/10.1109/HPCC/SmartCity/DSS.2019.00182

[11] W. K. Hastings. 1970. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika* 57 (1970), 97–109. https://api.semanticscholar.org/CorpusID:21204149

[12] Thomas P. Hayes and Alistair Sinclair. 2005. A general lower bound for mixing of single-site dynamics on graphs. *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)* (2005), 511–520. https://api.semanticscholar.org/CorpusID:15273753

[13] Yongchao He and Wenfei Wu. 2019. Fully Functional Rate Limiter Design on Programmable Hardware Switches. In *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos (SIGCOMM Posters and Demos '19)*. Association for Computing Machinery, New York, NY, USA, 159–160. https://doi.org/10.1145/3342280.3342344

[14] Paul Gerhard Hoel, Sidney C. Port, C. J. Stone, and Richard Holley. 1972. Introduction to Stochastic Processes. https://api.semanticscholar.org/CorpusID:120660908

[15] Richard Holley and Daniel Stroock. 1988. Simulated annealing via Sobolev inequalities. *Communications in Mathematical Physics* 115, 4 (1988), 553 – 569.

[16] D. Jerison. 2013. General mixing time bounds for finite Markov chains via the absolute spectral gap. *arXiv: Probability* (2013). https://api.semanticscholar.org/CorpusID:119324192

[17] Marcus Kaiser, Robert L. Jack, and Johannes Zimmer. 2016. Acceleration of Convergence to Equilibrium in Markov Chains by Breaking Detailed Balance. *Journal of Statistical Physics* 168 (2016), 259 – 287. https://api.semanticscholar.org/CorpusID:55280487

[18] Tony Lelièvre, Francis Nier, and Grigorios A. Pavliotis. 2012. Optimal Nonreversible Linear Drift for the Convergence to Equilibrium of a Diffusion. *Journal of Statistical Physics* 152 (2012), 237–274. https://api.semanticscholar.org/CorpusID:55349460

[19] László Miklós Lovász and Peter Winkler. 1995. Efficient stopping rules for Markov chains. In *Symposium on the Theory of Computing*. https://api.semanticscholar.org/CorpusID:14945719

[20] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics* 21, 6 (06 1953), 1087–1092. https://doi.org/10.1063/1.1699114 arXiv:https://pubs.aip.org/aip/jcp/article-pdf/21/6/1087/18802390/1087_1_online.pdf

[21] Milena Mihail. 1989. Conductance and convergence of Markov chains-a combinatorial treatment of expanders. *30th Annual Symposium on Foundations of Computer Science* (1989), 526–531. https://api.semanticscholar.org/CorpusID:7862836

[22] Barath Raghavan, Kashi Vishwanath, Sriram Ramabhadran, Kenneth Yocum, and Alex C. Snoeren. 2007. Cloud control with distributed rate limiting. In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '07)*. Association for Computing Machinery, New York, NY, USA, 337–348. https://doi.org/10.1145/1282380.1282419

[23] Luc Rey-Bellet and Konstantinos V. Spiliopoulos. 2014. Irreversible Langevin samplers and variance reduction: a large deviations approach. *Nonlinearity* 28 (2014), 2081 – 2103. https://api.semanticscholar.org/CorpusID:118327959

[24] Luc Rey-Bellet and Konstantinos V. Spiliopoulos. 2016. Improving the Convergence of Reversible Samplers. *Journal of Statistical Physics* 164 (2016), 472–494. https://api.semanticscholar.org/CorpusID:88514980

[25] Rade Stanojevi and Robert Shorten. 2008. Fully decentralized emulation of best-effort and processor sharing queues. *SIGMETRICS Perform. Eval. Rev.* 36, 1 (jun 2008), 383–394. https://doi.org/10.1145/1384529.1375501

[26] Puqi Perry Tang and T.-Y.C. Tai. 1999. Network traffic characterization using token bucket model. In *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, Vol. 1. 51–62 vol.1. https://doi.org/10.1109/INFCOM.1999.749252

[27] E. Vigoda. 1999. Improved bounds for sampling colorings. In *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*. 51–59. https://doi.org/10.1109/SFFCS.1999.814577

A LEMMAS AND PROOFS

Appendices are supporting material that has not been peer-reviewed.

This section lists all the lemmas referenced in this paper, among which Lemma A.1 and Lemma A.3 are well-known theorems in algebra and stochastic processes. The proof is omitted here and can be found in detail in reference [14].

Lemma A.1 (Perron-Frobenius Theorem). *Suppose $A \in \mathbb{R}^{n\times n}$ is non-negative and regular, i.e., A^k is elementwise positive for some $k \geq 1$, then*
(1) There is an eigenvalue λ_{pf} of A that is real and positive, with positive left and right eigenvectors.
(2) For any other eigenvalue λ, we have $|\lambda| < \lambda_{pf}$.
(3) The eigenvalue λ_{pf} is simple.
Suppose $D = Q^{-1}AQ$ is the Jordan Form of A, then the first row of Q^{-1} is the unique invariant probability vector π, and the elements of the first column of Q is 1. D is a block diagonal matrix, i.e., $D = diag(1, M)$, where $M^n \to 0$ $(n \to \infty)$, and

$$\lim_{n\to\infty} P^n = \lim_{n\to\infty} QD^nQ^{-1} = (\pi, ..., \pi)^T \tag{7}$$

Lemma A.2 (elementwise positive matrix). *Let $D_1 = diag(d_1, ..., d_p) \in \mathbb{R}^{p\times p}$, $D_2 = diag(d_{p+1}, ..., d_q) \in \mathbb{R}^{q\times q}$ be elementwise positive matrice, i.e., $d_i > 0$ $(i = 1, ..., q)$. Let $B = (b_{ij})_{p\times q}$ and $C = (c_{ij})_{q\times p}$ be elementwise positive matrice, i.e., $b_{ij} > 0$ and $c_{ij} > 0$. Suppose $P = \begin{pmatrix} D_1 & B \\ C & D_2 \end{pmatrix}$, then P^2 is an elementwise positive matrix.*

Proof. We calculate $P^2 = \begin{pmatrix} D_1^2 + BC & D_1B + BD_2 \\ CD_1 + D_2C & CB + D_2^2 \end{pmatrix}$, where $D_1^2 + BC$ and $CB + D_2^2$ are elementwise positive. Notice that $D_1B = (d_ib_{ij})_{p\times q}$ is elementwise positive, so every block of P^2 is elementwise positive. □

Lemma A.3 (Convergence Theorem). *Suppose that stochastic matrix P is irreducible and aperiodic, with stationary distribution π. Then there exist constants $0 < \alpha < 1$ and $C > 0$ such that*

$$\max_{x\in X} \|P^t(x, \cdot) - \pi\|_{TV} \leq C\alpha^t \tag{8}$$

Lemma A.4 (Bounding $d(t)$). *Given the Markov Chain with the transition matrix P, the Markovian coupling $\{(X_t, Y_t)\}_{t\geq 0}$ with $X_0 = x$ and $Y_0 = y$ is defined by*

$$\text{if } X_s = Y_s, \text{ then } X_t = Y_t \text{ for } t \geq s \tag{9}$$

Define coalescence time as

$$\tau_{couple} = \min\{t : X_s = Y_s \text{ for all } s \geq t\} \tag{10}$$

Define mixing distance as

$$d(t) = \max_{x \in \mathcal{X}} \|P^t(x, \cdot) - \pi\|_{TV} \tag{11}$$

Then $d(t)$ satisfies the following inequality:

$$d(t) \leq \max_{x,y \in \mathcal{X}} \mathbb{P}\{\tau_{couple} > t\} \tag{12}$$

Proof. Notice that $P^t(x, z) = \mathbb{P}\{X_t = z\}$, $P_t(y, z) = \mathbb{P}\{Y_t = z\}$, thus (X_t, Y_t) is a coupling of $P^t(x, \cdot)$ with $P^t(y, \cdot)$. From the basic upper bounding of the Total Variation, we have

$$\|P_t(x, \cdot) - P_t(y, \cdot)\|_{TV} \leq \mathbb{P}\{X_t \neq Y_t\} = \mathbb{P}\{\tau_{couple} > t\}$$

For any given set A,

$$\begin{aligned} |P^t(x, A) - \pi(A)| &= \left| \sum_y \pi(y)(P^t(x, A) - P^t(y, A)) \right| \\ &\leq \sum_y \pi(y) \|P^t(x, \cdot) - P^t(y, \cdot)\|_{TV} \\ &\leq \max_{x,y} \|P^t(x, \cdot) - P^t(y, \cdot)\|_{TV} \end{aligned}$$

Thus, $d(t) \leq \max_{x,y} \mathbb{P}\{\tau_{couple} > t\}$. □

Lemma A.5 (Bounding Convergence Speed using λ^*). *Let A denote a symmetric stochastic matrix, and let $\lambda^* = \max\{|\lambda| : \lambda \in \lambda(A), \lambda \neq 1\}$, then for all $k \geq 1$*

$$\|\mu A^k - \pi\|_{L^2} \leq \lambda^{*k} \|\mu - \pi\|_{L^2} \tag{13}$$

where μ is an random initial vector, and π is the stationary vector of A such that $\pi = \pi A$.

Proof. Let $V_2 = \text{span}(v_2, ..., v_n)$ be the $(n-1)$-dimensional space spanned by the eigenvectors corresponding to $\lambda_2, ..., \lambda_n$. In accordance with the properties of symmetric matrices, $v_1 \perp V_2$. Suppose $\mu = \pi + \omega$, where $\omega \in V_2$, then

$$\|\mu e^A - \pi\|_{L^2} = \|(\mu - \pi)e^A\|_{L^2} \leq e^{\lambda_2} \|\mu - \pi\|_{L^2}$$

The equivalenve of norm definition ensures that the assessment of convergence speed using $\|\mu - \pi\|_{L^2}$ is consistent with that using $\|\mu - \pi\|_{TV}$. □

Lemma A.6 (TV Distance and L1-Norm). *The TV Distance between two distribution u and v satisfies*

$$\|u - v\|_{TV} = \frac{1}{2} \sum_{x \in \mathcal{X}} |u(x) - v(x)| \tag{14}$$

Proof. Let $B = \{x : u(x) \geq v(x)\}$, $A \in \mathcal{X}$, then

$$\begin{aligned} u(A) - v(A) &\leq u(A \cup B) - v(A \cup B) \\ &\leq u(B) - v(B) \\ &\leq u(B^c) - v(B^c) \end{aligned}$$

Note that the right side of the equality is the upper bound of $|v(A) - u(A)|$. Let $A = B^c$, then $|v(A) - u(A)|$ equals to the upper bound, and we get

$$\|u - v\|_{TV} = \frac{1}{2}(u(B) - v(B) + v(B^c) - u(B^c)) = \frac{1}{2} \sum_{x \in \mathcal{X}} |u(x) - v(x)|$$

□

B INTRODUCTION TO MH ALGORITHM

We initialize an appropriate matrix $\hat{P}$ to accommodate specific topologies. For instance, in the matrix corresponding to a cycle topology, only the entries adjacent to the diagonal at the i^{th} row are non-zero, with the rest being zero. We then employ the MH algorithm to modify the preliminary matrix $\hat{P}$ to construct the matrix P. Let

$$P = \begin{cases} \hat{P}_{ij} \min\left\{1, \frac{\pi_j \hat{P}_{ji}}{\pi_i \hat{P}_{ij}}\right\} & i \neq j \\ \hat{P}_{ii} + \sum_{k \neq i} \hat{P}_{ik} \left(1 - \min\left\{1, \frac{\pi_k \hat{P}_{ki}}{\pi_i \hat{P}_{ik}}\right\}\right) & i = j \end{cases} \tag{15}$$

In fact, after calculating P_{ij} $(i \neq j)$, the distributed node i calculate $P_{ii} = 1 - \sum_{j \neq i} P_{ij}$.

The matrix P characterizes a reversible Markov Chain with a stationary distribution π. Given that various initial matrices $\hat{P}$ may lead to different constructions of P, verifying the existence of a limiting distribution by the criteria specified in Section 3.2 is essential.

C BOUNDING THE MIXING TIME FOR CYCLES

Take the cycle topology as an example. The graph G has vertex set $\{1, 2, ..., n\}$. Let D_t be the clock-wise distance from X_t to Y_t, and $\tau = \min\{t \geq 0 : D_t \in \{0, n\}\}$. Note that the process D_t is a simple random walk on the interior vertices of $\{0, 1, ..., n\}$ and gets absorbed at either 0 or n [14].

Assume that $P_i\{X_\tau = n\} = P_i$ is the probability that the walker is absorbed at n or 0, given it starts at the node k. We solve simultaneously for $P_0, ..., P_n$. We simplify the problem by assuming that all nodes have the same probability of transitioning to the left and the right, denoted as p and q respectively. Clearly $P_0 = 0$ and $P_n = 1$, while

$$P_k = pP_{k-1} + qP_{k+1} \ (1 \leq k \leq n-1) \tag{16}$$

Similarly, $\mathbb{E}_0\tau = \mathbb{E}_1\tau = 0$ while

$$\mathbb{E}_k\tau = p(1 + \mathbb{E}_{k+1}\tau) + q(1 + \mathbb{E}_{k-1}\tau) \ (1 \leq k \leq n-1) \tag{17}$$

The Equation 16 can be solved using the characteristic root method, and Equation 16 can be transformed into the form of Equation 16 by substitution $\mathbb{E}_k\tau = b_k + \frac{k}{q-p}$.

Let $x \neq 1$ be the root of $x = px^2 + q$. Solve the equation and we get

$$\mathbb{E}\tau = \begin{cases} \frac{1}{p-q} \frac{1-x^k}{1-x^n} + \frac{k}{p-q} & p > q \\ k(n-k) & p = q = \frac{1}{2} \end{cases} \tag{18}$$

Note that $\tau = \tau_{couple}$ as defined in Lemma A.4, and from Lemma A.4

$$d(t) \leq \max_{x,y} \mathbb{P}\{\tau > t\} \leq \frac{\max_{x,y} \mathbb{E}\tau}{t} \tag{19}$$

Let $d(t) = \epsilon$, then $\epsilon = d(t) \leq \frac{max_{x,y}\mathbb{E}\tau}{t}$, and

$$t_{mix}(\epsilon) \leq \begin{cases} \frac{n}{(p-q)\epsilon} & p > q \\ \frac{n^2}{4\epsilon} & p = q = \frac{1}{2} \end{cases} \tag{20}$$

For example, if $n = 10$, then $t_{mix}(0.05) \leq 50$.