Copula Theory and Its Applications in Computer Networks

by

Fang Dong
B.Sc., Wuhan University, 2011
M.Eng., Wuhan University, 2013

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Copula Theory and Its Applications in Computer Networks

by

Fang Dong
B.Sc., Wuhan University, 2011
M.Eng., Wuhan University, 2013

Supervisory Committee

Dr. Kui Wu, Co-Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Co-Supervisor
(Department of Computer Science)

Dr. Lin Cai, Outside Member
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. Kui Wu, Co-Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Co-Supervisor
(Department of Computer Science)

Dr. Lin Cai, Outside Member
(Department of Electrical and Computer Engineering)

ABSTRACT

Traffic modeling in computer networks has been researched for decades. A good model should reflect the features of real-world network traffic. With a good model, synthetic traffic data can be generated for experimental studies; network performance can be analysed mathematically; service provisioning and scheduling can be designed aligning with traffic changes. An important part of traffic modeling is to capture the dependence, either the dependence among different traffic flows or the temporal dependence within the same traffic flow. Nevertheless, the power of dependence models, especially those that capture the functional dependence, has not been fully explored in the domain of computer networks.

This thesis studies copula theory, a theory to describe dependence between random variables, and applies it for better performance evaluation and network resource provisioning. We apply copula to model both contemporaneous dependence between traffic flows and temporal dependence within the same flow. The dependence models are powerful and capture the functional dependence beyond the linear scope. With numerical examples, real-world experiments and simulations, we show that copula modeling can benefit many applications in computer networks, including, for example, tightening performance bounds in statistical network calculus, capturing full

dependence structure in Markov Modulated Poisson Process (MMPP), MMPP parameter estimation, and predictive resource provisioning for cloud-based composite services.

Contents

List of Tables

List of Figures

Nomenclature

Notation of Chapter 2

C	Copula
$C(u, v; \theta)$	Parametric copula
C_{lb}	Fréchet-Hoeffding lower bound copula
C_{ub}	Fréchet-Hoeffding upper bound copula
C_{ind}	Product copula
$\hat{C}$	Empirical copula
u, v	The argument value of copula, or the sample value of marginal distribution function
U, V, X, Y	Random variables
x, y	Sample value of random variables
F	Cumulative distribution function
$\hat{F}$	Empirical cumulative distribution function
ρ_τ	Kendall's tau
ρ_s	Spearman's rho
ρ	Pearson correlation coefficient
ρ_t^+	Upper tail dependence
ρ_t^-	Lower tail dependence

Notation of Chapter 3

$A(t)$	Cumulative traffic arrives in time interval $(0, t]$
$A^*(t)$	Cumulative traffic departs in time interval $(0, t]$
$S(t)$	Cumulative amount of service in time interval $(0, t]$
A	Traffic model
S	Service model
$\bar{F}$	Complementary distribution function/ survival function
α	The curve function in the definition of arrival model
β	The curve function in the definition of service model
Δ	A sliding window size
γ	Rate in SBB model
r_1, r_2	Parameter of exponential distributions
R_1, R_2	Constant service rate to flows
$\mathcal{B}(t)$	Backlog at time t
$\mathcal{D}$(t)	Delay at time t
$\mathbf{B}$	Random variable of backlog
$\mathbf{a}$	Random variable of the amount of data sent per unit of time
a^i	Sample value of $\mathbf{a}$ in the ith unit of time
$(\omega, \mu_1, \sigma_1, \mu_2, \sigma_2)$	Parameters of mixture of two Gaussian distributions

Notation of Chapter 4

(Q, Λ)	Parameter of MMPP
m	number of states in MMPP
Π	The stationary distribution for the CTMC

$P(t)$	The transition matrix for the CTMC after time t
I_i	i-th time slot
A_i	The random variable of the arrival count in i-th time slot of single MMPP trace
S_i	The random variable of the state of MMPP in i-th time slot
Δ	Length of time slots
M	The cumulative distribution function of A_i
$C_{i'}$	The copula between arrival counts A_i and $A_{i+i'}$, $i' \in \mathbb{N}$
G_j	The marginal distribution of A_i on the condition that associated CTMC is in state j
$\mathbb{G}(x)$	The vector $\mathbb{G}(x) = [G_1(x), G_2(x), \cdots, G_m(x)]$
A_i^l	The random variable of the arrival count in i-th time slot of HoMMPP/HeMMPP traces
M^l	The cumulative distribution function of A_i^l
$C_{i'}^l$	The copula between arrival counts A_i^l and $A_{i+i'}^l$, $i' \in \mathbb{N}$
$\nabla C_{i'}$	The single MMPP copula gardient
$\nabla C_{i'}^l$	The HoMMPP/HeMMPP copula gardient
$({}_lQ, {}_l\Lambda)$	The parameters of the l-th MMPP in HoMMPP/HeMMPP
${}_lA_i$	The random variable of the arrival count in i-th time slot of the l-th MMPP trace
${}_lM$	The cumulative distribution function of ${}_lA_i$
${}_lp$	The probability mass function of ${}_lA_i$
${}_lC_{i'}$	The copula between arrival counts ${}_lA_i$ and ${}_lA_{i+i'}$, $i' \in \mathbb{N}$
$\nabla {}_lC_{i'}$	The single MMPP copula gradient of the l-th MMPP

$\hat{a}$	The upper threshold of interested range of arrival counts
$\hat{M}^l$	The empirical cumulative distribution function of A_i^l
$C(u_i, u_{i+i'}; \theta)$	The parametric copula between A_i^l and $A_{i+i'}^l$ learnt from tarce

Notation of Chapter 5

x_i	Sample value of A_i or A_i^l
$\hat{x}_i$	Predicted value of A_i or A_i^l
$c(u_i, u_{i+i'}; \theta)$	Parametric copula density function
$(\varphi_1, \varphi_2, \epsilon_t)$	Parameters of AR(1) model
σ	Parameter of LPC(1) model
A_i'	An associate trace of A_i

Notation of Chapter 6

d	Scaling delay
β	Capacity unit
γ_j	Current capacity for VF j
μ_j	Capacity level for VF j
S_c	Copula-based scaling matrix
S_u	Utilization-based scaling matrix
S_g	Integrated scaling matrix
ϱ	Utilization of queueing system

Notation of Chapter 7

$u_i(\hat{u}_i)$	Marginal (empirical) distribution value of A_i
$\xi_i(\hat{\xi}_i)$	(Empirical) copula value of A_i and A_{i+1}
W_1, W_2	Objective function to minimize in two-step matching

Θ_1, Θ_2	Parameter sets to estimate in the first, and the second step
α	Step-size of gradient descent
$\Theta_1^{(r)}$, $\alpha^{(r)}$	Estimated parameter, step-size in the r-th iteration
H	Coefficient matrix for copula matching
E	Constraints coefficient matrix for copula matching
b	Constraints vector for copula matching
D_M	K-S distance between testing marginal and empirical marginal distributions
D_C	K-S distance between testing copula and empirical copula

ACKNOWLEDGEMENTS

I would like to thank:

my supervisors, Dr. Kui Wu and Dr. Venkatesh Srinivasan, for giving me the strong support and guidance during my PhD. Whenever I am stuck with a research problem or have questions about research, you are always open to help me. Your continuous advising and mentoring in the fast four years are of great value to me. I am deeply grateful and happy to pursue a PhD degree under your supervision.

my husband, Dr. Cheng Chen, for your love. You have always been with me through all those tough moments. Your encouragement always gives me the passion and strength to pursue what we believe and what we value the most. Your companionship makes our life wonderful and full of happiness.

my family, for your unconditional love and companionship. You are always there to share and witness every moment of my life even though we are not living in the same country. I feel sorry that we don't have much time together physically these years. I would like to express my sincere appreciation for your support and encouragement during the years of my education.

my labmates and friends, for sincere friendship, your valuable advice and help, and the unforgettable moments we have spent together.

DEDICATION

To my family.

Chapter 1

Introduction

In this chapter, we describe the motivation for applying copula theory in the computer network domain, and explain our research goals and contributions.

1.1 Motivation

In the modern society, our daily life heavily depends on computer networks. Everyday, tremendous network traffic is transmitted in both local area networks and the Internet for various applications. Whenever we transmit files between hosts, access a remote computer, visit a website, or watch a video online, network packets are generated and transmitted on networks. As more and more applications are emerging over the Internet, there is a high demand to explore accurate and robust models for network traffic flows.

In many cases, a good network traffic model is a prerequisite for research in computer networks. A good network traffic model means that the model can characterize and mimic specific real network traffic well. A good model can identify specific network traffic [57], simulate the traffic similar to the real traffic [48], and analyse the network performance [9].

Network traffic models can be divided into two groups: the models for statistical properties, such as mean, variance, skewness [21], and the models for dependence, such as covariance and correlation [46, 53]. The dependence modeling is of great significance to characterize network traffic and deepen our understanding of network traffic from a different angle. Considering the period of FIFA World Cup or Olympic Games, hundreds of thousands of people may visit the same website to watch the

game videos from home computers. When modeling the network traffic flows sent from home computers to the designed server, we cannot just add up the models of each individual flow, rather we need to take the dependence among the constituent flows into consideration. A dependence model between traffic flows will lead to a more accurate model for the aggregate flow and help to improve the analysis of network performance. In another example where there is a single traffic flow from a source to a destination, understanding the dependence between its arrivals over different times is important to predict future arrivals or detect abnormal events [4].

The two scenarios we consider above show the impact of two categories of dependence in network traffic, the contemporaneous dependence and the temporal dependence. The contemporaneous dependence is the dependence between arrivals from different traffic flows, while temporal dependence is the dependence between arrivals from the same traffic flow but over different times. Both contemporaneous and temporal dependencies in network traffic are non trivial to model. The contemporaneous dependence in network traffic is normally ignored for ease of analysis. Network performance analysis under stochastic network framework suffers from this ignorance and leads to a loose bound on network delay or backlog in practice [44]. The temporal dependence in one network traffic flow has existing solutions that are mostly based on the covariance or correlation [53]. However, the covariance or correlation can only measure the linear dependence, which discards abundant dependence information carried by traffic flows. We take the traffic trace BCpAug89 [32] as an example. Fig. 1.1 shows the scatter plot of the successive arrival counts (number of arrivals) every second. The shape of the scatter plot shows the dependence between successive arrival counts. From the figure, the linear dependence only considers the projection of all the points onto a straight line, while neglecting their (varying) vertical distances to the line. Therefore, linear dependence measures, such as covariance and autocorrelation, only measure the dependence partially, and are far from sufficient to reflect the complex dependence structure.

With the significance of network dependence modeling and the lack of rich models that capture the full spectrum of dependence structures, we are motivated to apply an advanced tool, copula, to model the functional dependence of network traffic and apply the new model to improve network studies. Copulas, as the term indicates, are functions that join one-dimensional marginal distributions to multivariate distributions. As an effective mathematical tool to capture dependence, copulas have been very popular in the domain of financial analysis, especially for risk manage-

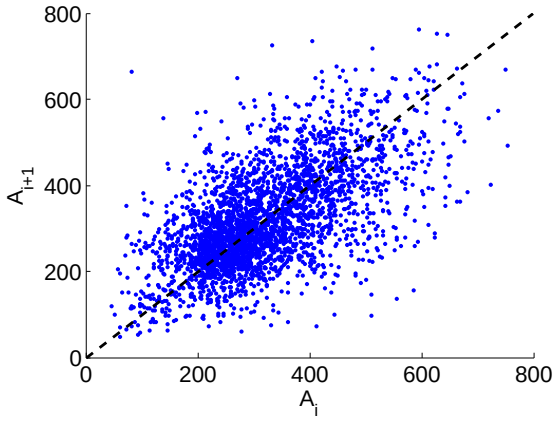


Figure 1.1: Scatter plot of successive arrival counts of BCpAug89

ment. To estimate the market risk appropriately, more than one assets need to be considered. Copulas are shown flexible and useful to measure the dependence between assets[67, 40] and the dependence along the time series of a single asset[66, 65, 71]. Although copulas have been considerably researched in the finance domain, they are quite new and rarely exploited in other domains. In recent years, researchers attempt to extend the usage of copulas in other areas. Specifically, copulas are used in the telecommunication networks domain to model the shortest-path trees[60], and in the agriculture domain to model the dependence between energy and agricultural commodities[50, 49]. To the best of our knowledge, copulas are seldom applied in computer networks domain, though dependence modeling of network traffic attracts a lot of attention and is considered of great significance for the examination and improvement of the network performance[46].

1.2 Research Goals

This thesis applies copulas to improve both contemporaneous and temporal dependence modeling of network traffic, which could further benefit the applications relying on dependence. Specifically, the research goals are described as follows:

1. **Contemporaneous dependence modeling**: Model the contemporaneous dependence between network traffic flows with copula. Contemporaneous dependence modeling is integrated to a network analysis framework, stochastic network calculus (SNC). With the contemporaneous dependence captured, the derived performance bounds would be tighter and more accurate.

2. **Temporal dependence modeling**: Model the temporal dependence for network traffic flow with copula. The temporal dependence in terms of copula can be used to improve the following network applications:

   - **Network traffic prediction**: By understanding the temporal dependence of network traffic, we can find a solution to predict the future arrivals based on current observations.
   - **Cloud service provisioning**: This application is based on network traffic predictions. Cloud service can be better offered according to the requested amount. Designing an effective service provisioning strategy based on prediction of requested amount is a goal in this context.
   - **Parameter estimation problem**: We propose a parameter estimation method for a widely-used network traffic model, Markov Modulated Poisson Process. The parameters will be estimated by matching statistical moments and temporal dependence, separately. We study both theoretical and parametric copulas for MMPP and design a method for fast and accurate parameter estimation.

1.3 Contributions

The thesis makes the following contributions:

1. **Copula analysis for contemporaneous dependence in statistical network calculus**
   In Chapter 3, we integrate copula into the framework of SNC and make the following contributions:

   - we augment the power of SNC with copula analysis to utilize the dependence structure between traffic flows. In particular, copula analysis can be integrated into the SNC framework to provide tighter performance bounds. Such analysis offers extra benefit in inferring the adaptive behavior of some proprietary systems.
   - Using copula analysis, we show the range of stochastic bounds that SNC can achieve. This discovery has a deep implication in the future design of flow scheduling or input buffering methods.

- A real-world case study as well as simulation evaluation demonstrate the practicality of copula analysis and its improvement over the performance of SNC that is oblivious to the dependence structures between flows.

2. **Copula analysis for temporal dependence of Markov Modulated Poisson Process**
   In Chapter 4, we fully study the temporal dependence of Markov Modulated Poisson Process and makes the following contributions:

   - We use copula to analyse the dependence structure of MMPP traffic. The copula-based dependence reveals richer information of temporal dependence and is more powerful than the commonly-used measures, covariance and correlation.
   - We give the exact form of temporal dependence of MMPP with arbitrary number of states. This is the first theoretical result on the functional temporal dependence of multi-state MMPP.
   - We propose a way to construct copula for superposition of MMPPs. Recursive algorithms are designed to calculate the numerical values of copulas.
   - We propose parametric copula modeling method for both single MMPP and superposition of MMPPs.

3. **Application of MMPP copula for traffic prediction**
   In Chapter 5, we apply MMPP copula for network traffic flow prediction and make the following contributions:

   - We introduce MMPP traffic prediction based on either theoretical copulas or parametric copulas.
   - We demonstrate applications of MMPP copula on both real-world traffic traces and simulated traffic traces. Both single MMPP flow and superposition of multiple MMPP flows are studied.
   - Case studies show that our copula-based traffic prediction method is more accurate and stable than existing methods.

4. **Application of MMPP copula in collaborative auto-scaling of cloud service**
   In Chapter 6, we apply MMPP copula in composite cloud service system to

design effective service provisioning strategy and make the following contributions:

- We introduce a novel approximation approach that transforms the time-ordered, spatially distributed calls to virtual functions (VFs) into a Markov Modulated Poisson Process (MMPP). This method solves the challenging problem in performance modeling of composite service, where the workflow of a task may pass through multiple VFs in an arbitrary order. By analysing the performance of MMPP input into a virtual queue, we can easily estimate the performance of composite services.
- To address the difficulty that the amount of calls at different VFs might scale up differently, we introduce a copula model to capture the stable dependence structure, even if the amount of calls to different VFs may scale up differently. This unique feature greatly simplifies the dependence modeling, since there is no need to rebuild the dependence model when the total amount of service calls varies.
- Cloud brokerage needs a mechanism to carefully balance the cost of purchasing VF resources and the QoS of composite service. As such, we propose a tiered, collaborative resource auto-scaling strategy, based on the predictive power of the copula model.

5. **Application of MMPP copula in parameter estimation**
   In Chapter 7, we apply MMPP copula to develop a fast and accurate estimation method to learn parameters of MMPP, and make the following contributions:

   - We model the joint behavior of successive arrival counts in terms of their marginal distribution and copula. The theoretical forms of marginal distribution and copula of arrival counts in MMPP lay solid foundation for parameter estimation.
   - Based on the MMPP copula, we propose a two-step estimation algorithm, MarCpa, to estimate MMPP parameters by matching marginal and matching copula separately.
   - Case studies with a large number of simulations demonstrate that our proposed method is more efficient and accurate than existing estimation methods that learn MMPP parameters from arrival counts.

1.4 Publications

Fang Dong, Kui Wu, and Venkatesh Srinivasan. "Copula Analysis for Statistical Network Calculus," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, April 2015.

Fang Dong, Kui Wu, Venkatesh Srinivasan, and Jianping Wang. "Copula Analysis of Latent Dependency Structure for Collaborative Auto-scaling of Cloud Services", in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, August 2016.

Fang Dong, Kui Wu, Venkatesh Srinivasan. "Copula-based Parameter Estimation for Markov-modulated Poisson Process", in *Proceedings of IEEE/ACM International Symposium on Quality of Service (IWQoS)*, June 2017.

Fang Dong, Kui Wu, Venkatesh Srinivasan. "Copula Analysis of Temporal Dependence Structure in Markov Modulated Poisson Process and Its Applications," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (ToMPECS)*, accepted in May 2017.

Chapter 2

Preliminaries on Copula Theory

2.1 Definitions and Basic Properties

We start with the definition of copulas and three core theorems.

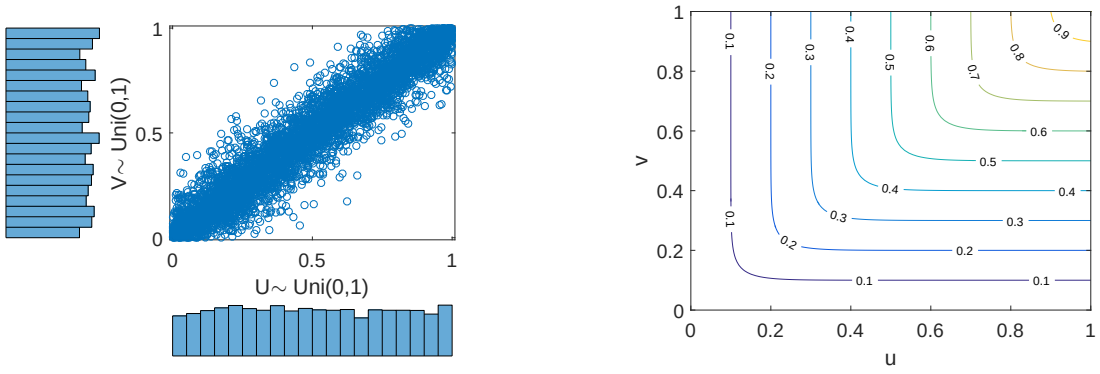
Definition 1. ***(Copulas)*** *A 2-dimensional copula is a function C having the following properties [59]:*

1. *Its domain is $[0,1] \times [0,1]$;*
2. *C is 2-increasing, i.e., for every $u_1, u_2, v_1, v_2 \in [0,1]$ and $u_1 \leq u_2, v_1 \leq v_2$, we have $C(u_2, v_2) - C(u_2, v_1) - C(u_1, v_2) + C(u_1, v_1) \geq 0$.*
3. *$C(u, 0) = C(0, v) = 0$, $C(u, 1) = u$, $C(1, v) = v$, for every $u, v \in [0,1]$.*

The function is called a subcopula if it has the second and the third properties of copula, but its domain is $b_1 \times b_2$, where b_1 and b_2 are subsets of $[0,1]$ containing 1 and 0.

By definition, a copula is essentially the joint distribution function of two random variables, denoted by U and V, that follow uniform distributions on the interval $[0,1]$. That is, $C(u,v) = F_{UV}(u,v)$ where $U \sim \text{Uni}(0,1), V \sim \text{Uni}(0,1)$ and F_{UV} is their joint distribution. An example is given in Example 1 to visualize the idea. In the example, the scatter plot shows the way U and V jointly distribute; In other words, the plot suggests the relationship between U and V. Different relationships will lead to different copulas. Therefore, the shape of a scatter plot of U and V indicates copula. Both scatter plot and contour are widely-used ways to visualize a copula.

Example 1. *Consider two random variables U and V that follow uniform distribution on $[0, 1]$ and their samples shown in scatter plot in Fig. 2.1a. The scatter plot shows how U and V jointly distribute on two dimensional plane. The contour of the related copula is shown in Fig. 2.1b.*



(a) Scatter plot of samples of (U, V) (b) Contour of the related copula

Figure 2.1: An explanatory example of the definition of copula.

Theorem 1. ***(Sklar's theorem)*** *[59] Let F_{XY} be a joint distribution function with marginals F_X and F_Y, then there exists a copula C such that for for all x and y, $F_{XY}(x, y) = Pr(X \leq x, Y \leq y) = C(F_X(x), F_Y(y))$.*

If the marginals F_X and F_Y are continuous, then copula C is unique; otherwise, C is uniquely determined on the range of the marginals. Example 2 is given for explanation of the theorem. Sklars theorem is the core of copula theory. It shows how copula connects marginals with joint distribution, which is the essential way that copula captures dependence between random variables. On one hand, Sklars theorem is especially useful since the joint distribution of random variables is hard to find directly in many applications [11, 59]. In this situation, integration of a copula model and marginals makes it easy to understand the joint behaviour. On the other hand, Sklar's theorem implies that copula, as a dependence measure, is entirely separated from both marginals and joint distribution. The modeling of marginal distributions and the modeling of copula could be totally separate to fit different application scenarios.

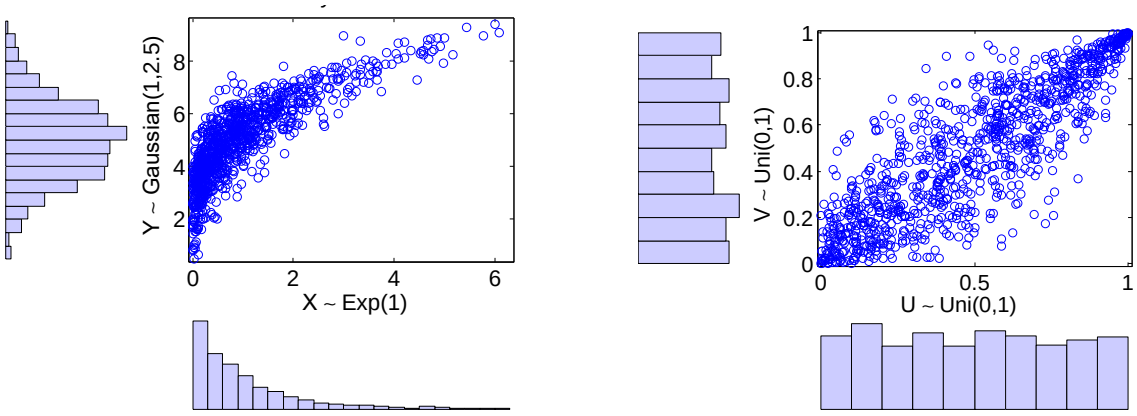
Example 2. *Consider two random variables $X \sim Exp(1)$ and $Y \sim Gaussian(1, 2.5)$, with their samples (x, y) shown in Fig. 2.2a. Regarding the marginal distribution value*

of X and Y as random variable U and V, every sample pair (x, y) is mapped to a sample pair (u, v) in the marginal domain in the way

$$u = F_X(x) = Pr(X \leq x) = 1 - e^{-x},$$
$$v = F_Y(y) = Pr(Y \leq y) = \frac{1}{2.5\sqrt{2\pi}} \int_{-\infty}^{y} \frac{-(y'-1)^2}{2 * 2.5^2} dy'.$$

The scatter plot of U and V in Fig. 2.2b indicates the copula that represents the joint distribution of U and V, and is called the copula between X and Y. The copula links the marginal distribution of X and Y into their joint distribution in the way

$$Pr(X \leq x, Y \leq y) = Pr(U \leq u, V \leq v) = C(u, v) = C(F(x), F(y)).$$



(a) X-Y scatter plot. (b) U-V scatter plot.

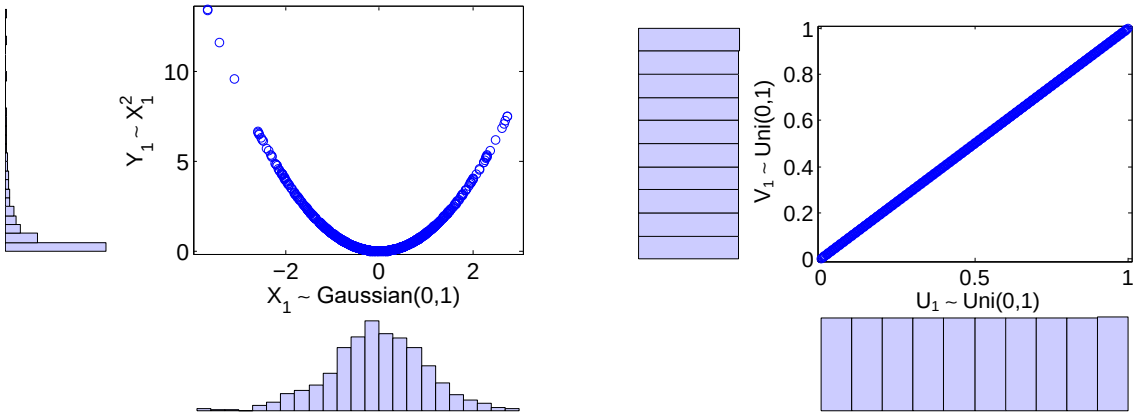
Figure 2.2: An explanatory example of Sklar's theorem.

Theorem 2. ***(The invariant property of copulas)*** *[59] Let X and Y be continuous random variables with copula C_{XY}. If α_1 and α_2 are strictly increasing functions on the range of X and the range of Y, respectively, then $C_{\alpha_1(X)\alpha_2(Y)} = C_{XY}$. In other words, C_{XY} is invariant under strictly increasing transformations of X and Y.*

As Sklar's theorem shows, copula is independent from both marginals and joint distributions, so the dependence in terms of copula is stable when the marginals change functionally, which is formally defined in the above invariant property. The practical meaning of the invariant property in computer networks domain is that the contemporaneous dependence between traffic flows and the temporal dependence

within one traffic flow in terms of copula will remain the same, even when the flow arrivals all scale up functionally. On this condition, we don't need to build the dependence repeatedly. Example 3 shows an example for the invariant property. The example also shows other dependence measures, such as correlation and covariance, don't satisfy the invariant property, making copula much more stable for practical use.

Example 3. *X_1 is a random variable Gaussian distributed with the mean as 0 and the standard deviation as 1. Y_1 is a random variable functionally dependent with X_1, i.e., $Y_1 = X_1^2$. Fig. 2.3a and 2.3b shows the scatter plot of X_1 and Y_1, and the scatter plot in the marginal domain, respectively.*



(a) X_1-Y_1 scatter plot.

(b) U_1-V_1 scatter plot.

(c) X_2-Y_2 scatter plot.

(d) U_2-V_2 scatter plot.

Figure 2.3: An explanatory example of the invariant property

Let's generate another two random variables by applying increasing functions on X_1 and Y_1 respectively, e.g., $X_2 = X_1^2$, $Y_2 = Y_1 + 3$. After the transformation, the

$X_2 - Y_2$ scatter plot, in Fig. 2.3c, appears completely different from $X_1 - Y_1$ scatter plot. However, in the marginal domain, the scatter plot turns to be the same as comparing Fig. 2.3d and 2.3b. As the scatter plot figures of $U_1 - V_1$ and $U_2 - V_2$ indicate two copulas, we can tell the dependence structure between random variables, in terms of copulas, has been kept stable under the increasing function transformation. From Figs. 2.3a and 2.3c, we can also tell that X_1 and Y_1 are not linearly dependent, whereas X_2 and Y_2 are. Therefore, the linear dependence structure is not invariant under functional transformation.

Theorem 3. ***(Fréchet-Hoeffding bounds)*** *[59] For every copula C and for all u, v in $[0, 1]$, the following inequality holds*

$$C_{lb}(u, v) = max(u + v - 1, 0) \leq C(u, v) \leq min(u, v) = C_{ub}(u, v). \tag{2.1}$$

We refer to C_{ub} as the Fréchet-Hoeffding upper bound and C_{lb} as the Fréchet-Hoeffding lower bound.

Fréchet-Hoeffding bounds show the range of all possible copulas. Consider copula C to model the dependence between X and Y. When $C = C_{lb}$, Y is a decreasing function of X; when $C = C_{ub}$, Y is an increasing function of X[35]. Therefore Fréchet-Hoeffding bounds actually capture two extreme functional dependencies. Except for these two special copulas, a third important copula is product copula, $C_{ind}(u, v) = uv$. X and Y is independent if their copula is C_{ind}. Figs. 2.4, 2.5 and 2.6 visualize the copulas C_{lb}, C_{ind} and C_{ub}, respectively, with their scatter plot figures and contour figures.

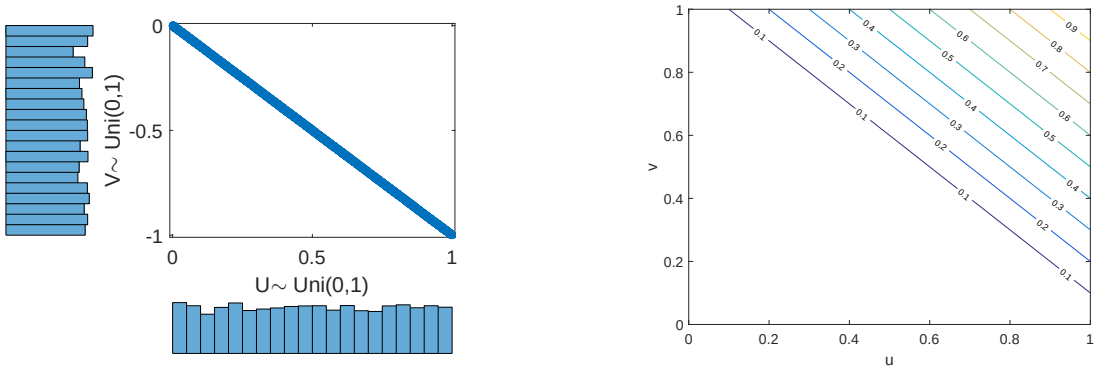
Theorem 4. ***(Inversion method)*** *[59] Let F_{XY} be a joint distribution function with marginals F_X and F_Y. Let F_X^{-1} and F_Y^{-1} be the inverse function of F_X and F_Y. Then the copula between X and Y can be constructed as*

$$C(u, v) = F_{XY}(F_X^{-1}(u), F_Y^{-1}(v)) \quad \forall u, v,$$

such that

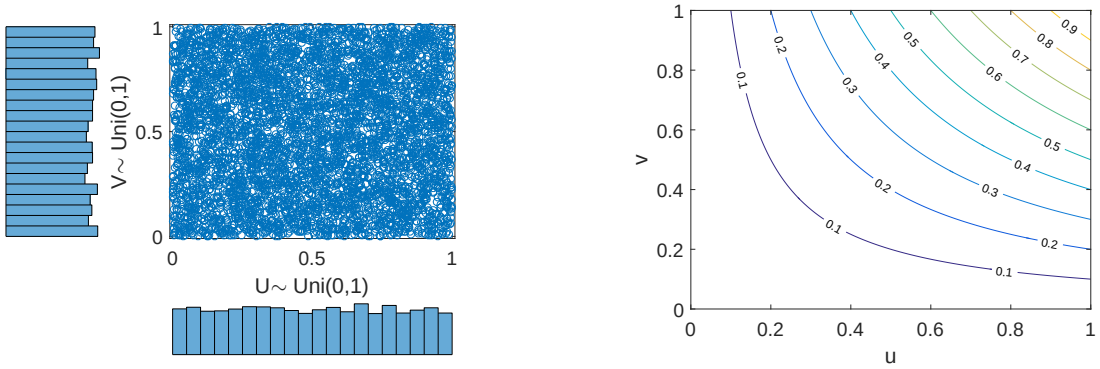
$$F_{XY}(x, y) = C(F_X(x), F_Y(y)) \quad \forall x, y.$$

The inversion method is used to construct a theoretical copula for the problem at hand. It uses Sklar's theorem to construct copulas. The inversion method leads



(a) Scatter plot in $U - V$ plane. (b) Copula contour.

Figure 2.4: Fréchet-Hoeffding lower bound copula C_{lb}.



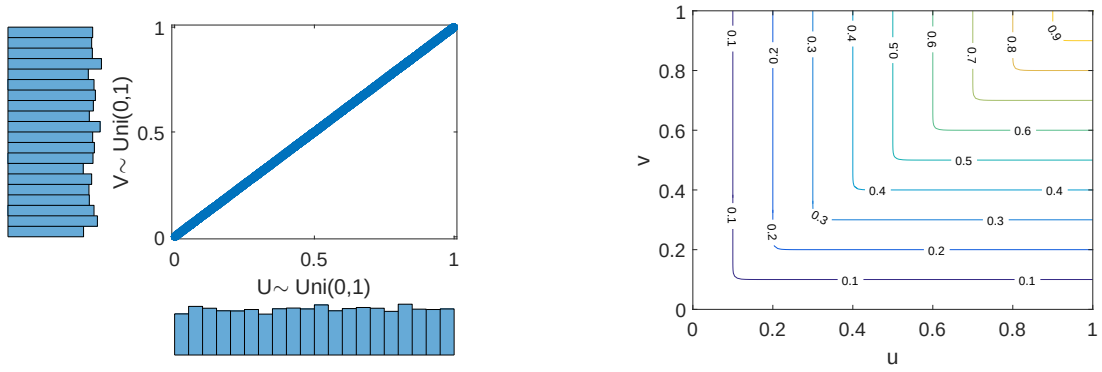
(a) Scatter plot in $U - V$ plane. (b) Copula contour.

Figure 2.5: Product copula C_{ind}.

to a unique copula when the marginals are continuous, and leads to a unique subcopula when the marginals are not continuous. The unique subcopula can be easily extended to a copula via various ways, for instance, bilinear interpolation [59]. Thus, a subcopula shares most properties of copulas. In the following context, we do not differentiate between subcopula and copula, because their difference does not impact the our analysis and application in following chapters.

2.2 Copula-based Dependence Measures

The copula-based dependence measures satisfy the invariant property as shown in Theorem 2. There are two main ways to measure the copula-based dependence. One is based on concordance statistics, which measures the extent to which two random



(a) Scatter plot in $U - V$ plane.

(b) Copula contour.

Figure 2.6: Fréchet-Hoeffding upper bound copula C_{ub}.

variables are both large or small at the same time. The other one is tail dependence, which measures the amount of dependence in the upper and lower quadrant tail of joint distributions.

Kendall's tau and Spearman's rho are two popular copula-based dependence measures defined in terms of concordance. Their definitions are as follows:

Definition 2. ***(Kendall's tau)*** *[59] Let (X_i, Y_i) and (X_j, Y_j) denote two observations from a vector (X, Y) of continuous random variables with copula between X and Y as $C(u, v)$, the Kendalls' tau is defined as*

$$\rho_\tau = Pr((X_i - X_j)(Y_i - Y_j) > 0) - Pr((X_i - X_j)(Y_i - Y_j) < 0) = 4 \int_0^1 \int_0^1 C(u, v) dC(u, v) - 1. \tag{2.2}$$

Definition 3. ***(Spearman's rho)*** *[59] Let (X_i, Y_i), (X_j, Y_j) and (X_k, Y_k) denote three observations from a vector (X, Y) of continuous random variables with copula between them as $C(u, v)$, the Spearman's rho is defined as*

$$\rho_s = 3(Pr((X_i - X_j)(Y_i - Y_k) > 0) - Pr((X_i - X_j)(Y_i - Y_k) < 0)) = 12 \int_0^1 \int_0^1 C(u, v) du dv - 3. \tag{2.3}$$

Essentially, both Kendall's tau and Spearman's rho are calculated by using concordance minus discordance between samples of two random variables. Although their values could be quite different, they have the same range from 0 to 1, and are monotonic increasing functions of each other. From the values of Kendall's tau

and Spearman's rho, the degree of dependence is explained as follows: a large value indicates stronger positive functional dependence between variables, and a smaller value indicates stronger negative functional dependence between variables. The functional dependence degree is reflected by absolute values $|\rho_\tau|$ or $|\rho_s|$. Three special dependence values are listed below with the related copulas [29]:

- $\rho_\tau = 1$ or $\rho_s = 1$ is equivalent to $C = C_{ub}$, indicating the largest positive functional dependence;
- $\rho_\tau = -1$ or $\rho_s = -1$ is equivalent to $C = C_{lb}$, indicating the largest negative functional dependence;
- $\rho_\tau = 0$ or $\rho_s = 0$ is equivalent to $C = C_{ind}$, indicating the independence

As copula-based measures, both Kendall's tau and Spearman's rho can capture dependence beyond linear scope. Taking X_1 and Y_1 in Example 3 as an example, the copula between X_1 and Y_1 is C_{ub}, and the copula-based dependence degree between the two random variables are $\rho_\tau = 1$ and $\rho_s = 1$. With copula-based dependence measures, the strong functional dependence between X_1 and Y_1 has been shown. However, with linear dependence measures, for example, Pearson correlation coefficient, $\rho = 0$ between X_1 and Y_1 shows a zero dependence degree, and does not reflect the actual dependence.

Tail dependence calculates the probability that two random variables achieve extreme large (or small) value simultaneously. The upper tail dependence and lower tail dependence are defined as follows:

Definition 4. ***(Tail dependence)*** *[29] Given two random variables X and Y with marginals as F_X and F_Y, and their copula C, the upper tail dependence is*

$$\rho_t^+ = \lim_{u \to 1} Pr(X > F_X^{-1}(u) | Y > F_Y^{-1}(u)) = \lim_{u \to 1} \frac{1 - 2u + C(u, v)}{1 - u}; \tag{2.4}$$

the lower tail dependence is

$$\rho_t^- = \lim_{u \to 0} Pr(X < F^{-1}(u) | Y < F^{-1}(u)) = \lim_{u \to 0} \frac{C(u, u)}{u}. \tag{2.5}$$

In practice, the tail dependence shows the possibility of the concurrence of two extreme events. The information on the concurrence of extreme events gives a new aspect of understanding of dependence, and is helpful to monitor and identify events on extreme conditions.

2.3 Parametric Copulas

In many applications, the exact copulas between random variables are difficult to construct. So parametric families of copulas have been proposed and explored to cover various types of dependence structures. Elliptical copulas and Archimedean copulas are two copula families mostly studied. Elliptical copulas are derived from multivariate distribution implicitly. They strictly have symmetrical lower tail dependence and upper tail dependence, indicating that the probability of occurrence of extreme large values is equal to the probability of occurrence of extreme small values. The typical elliptical copulas are Gaussian copula and Student's t copula.

Archimedean copulas are explicit copulas, which have clear and closed forms. Compared with elliptical copulas, Archimedean copulas are more flexible on the property of tail dependence. They could model either equal or distinct upper and lower tail dependence. Besides, Archimedean copulas are easier to construct due to the few parameters to estimate. Even with few parameters, this family of copulas include a great variety of copulas, and can model the dependence structure very effectively. All these advantages make Archimedean copulas good candidates for most applications. Three popular one-parameter Archimedean copulas are Clayton copula, Gumbel copula and Frank copula:

- Clayton copula

$$C(u, v; \theta) = [\max\{u^{-\theta} + v^{-\theta} - 1, 0\}]^{-1/\theta}, \quad \theta \in [-1, \infty) \setminus \{0\};$$

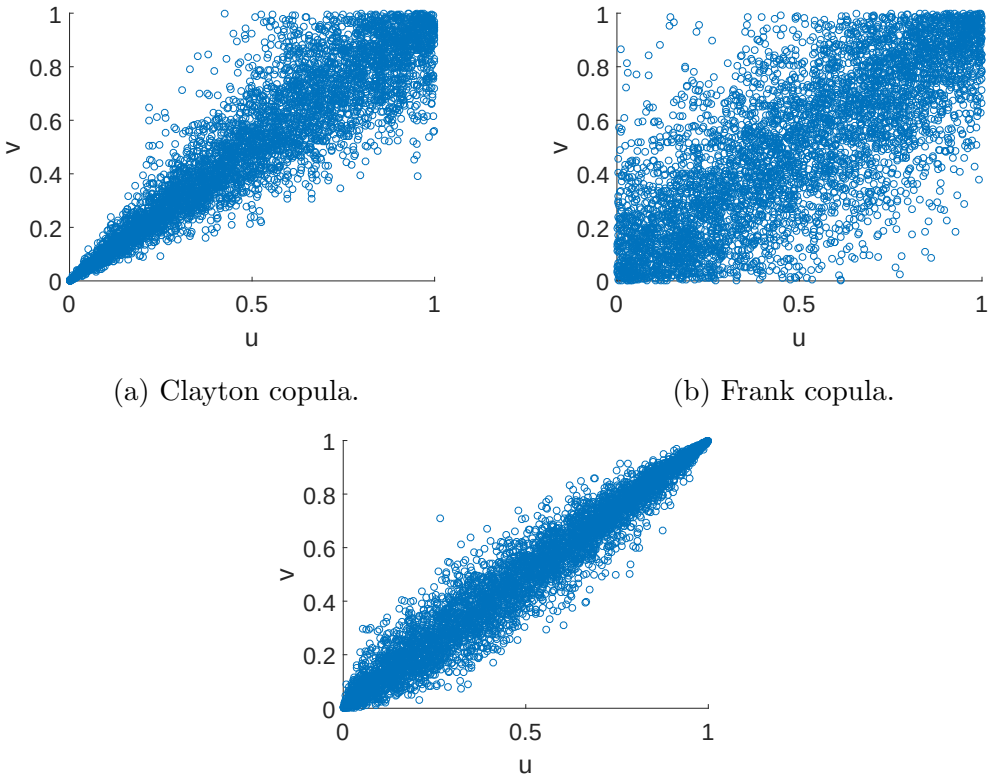
- Frank copula

$$C(u, v; \theta) = -\frac{1}{\theta} \log[1 + \frac{(\exp(-\theta u) - 1)(\exp(-\theta v) - 1)}{\exp(-\theta) - 1}], \quad \theta \in [-\infty, \infty) \setminus \{0\};$$

- Gumbel copula

$$C(u, v; \theta) = \exp[-((-\log u)^{\theta} + (-\log v)^{\theta})^{1/\theta}], \quad \theta \in [1, \infty).$$

The scatter plot figures of these three copulas are shown in Fig. 2.7. The three copulas are widely used due to several reasons. First, they are all one-parameter copulas, making it easier to fit models into the real problem. Second, the parameter of copula relates to copula-based dependence, Kendall's tau and Spearman' rho directly.



(a) Clayton copula.

(b) Frank copula.

(c) Gumbel copula.

Figure 2.7: Scatter plot figures of three Archimedean copulas with parameter $\theta = 7$.

For instance, $\rho_\tau = \theta/(\theta+2)$ for Clayton copula, and $\rho_\tau = 1 - 1/\theta$ for Gumbel copula. Thus the copula parameter itself reflects the degree of dependence. Finally, the three copulas capture three extremely distinct tail dependencies. Specifically, Clayton copula captures low tail dependence, Gumbel copula captures upper tail dependence, and Frank copula capture symmetric tail dependence. Taking Clayton copula as an example, we can observe that samples cluster on the bottom left of scatter plot in Fig. 2.7a, indicating strong lower tail dependence. In this thesis, we will exploit these three Archimedean copulas for dependence modeling in network applications.

2.4 Empirical Copula

Empirical copula is statistically counted from samples and defined as

Definition 5. *Given two random variables X and Y, and n number of observed*

sample pairs (x_i, y_i). *The empirical copula between* X *and* Y, $\hat{C}$ *is defined as:*

$$\hat{C}(u,v) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(u_i \leq u, v_i \leq v) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(\hat{F}_X(x_i) \leq u, \hat{F}_Y(y_i) \leq v), \tag{2.6}$$

where $\hat{F}_X$ *and* $\hat{F}_Y$ *are empirical marginal distribution functions defined as*

$$\hat{F}_X(x_i) = \frac{1}{n}\sum_{i'=1}^{n} \mathbf{1}(x_{i'} \leq x_i) \tag{2.7}$$

$$\hat{F}_Y(y_i) = \frac{1}{n}\sum_{i'=1}^{n} \mathbf{1}(y_{i'} \leq y_i) \tag{2.8}$$

From the definition, the empirical copula is purely determined by samples, so it is the raw model that represents the samples. Empirical copula can be used as benchmark to test whether a parametric copula is the underlying copula of samples [36]. Many research works use empirical copula for goodness-of-fitting test [26, 36]. If the parametric copula to test is close enough to empirical copula, it can be accepted as the underlying copula; otherwise, it is not the copula of samples.

2.5 Summary

From the introduction to copula theory in this chapter, we show the advantages of copulas for dependence modeling. First, copulas can measure the functional dependence beyond linear scope with Spearman's rho and Kendall's tau. Second, copulas separate marginals from joint distributions, allowing copulas to remain stable and invariant even when the marginals change functionally. Third, copulas are very useful to reveal the joint information of random variables. Usually, marginals are more accessible than joint distributions, and joint distributions are hard to find directly. In this situation, integration of a copula model and marginals makes it easier to understand the joint behaviour. All these benefits of copulas help to better understand the dependence in network traffic. Therefore, in this thesis, we use copula theory for dependence modeling, and explore its applications in different computer network scenarios.

Chapter 3

Copula Analysis for Contemporaneous Dependence and Its Application in Statistical Network Calculus

3.1 Introduction

Since its introduction in early 1990s [22], network calculus has been widely adopted to analyse complex queueing systems, such as multimedia networks, where the Markovian property of arrivals generally does not hold and thus traditional queueing theory becomes hard to apply. Network calculus was initially developed along the deterministic track [15, 51] and later evolved to stochastic version [15, 18, 30, 44]. Stochastic network calculus (SNC) has received much attention in recent years due to its power in deriving probabilistic performance bounds, which are more meaningful in practice.

The practical use of SNC, however, has faced challenges due to the lingering problem in deriving tight stochastic performance bounds [20]. In particular, inappropriate traffic models and the extensive use of model transform may lead to loose performance bounds [20]. While substantial efforts have been devoted to improving the bounds [19, 42], the problem has only been tackled for special types of traffic and service models, using probability inequalities, *e.g.*, Chernoff bounds and martingale inequalities. In many cases, the independence assumption is required to ease the analysis, *e.g.*, the independence of the traffic arrivals and the independence between

the arrivals and the service.

In practice, loose bounds may occur due to the inaccurate *a-prior* traffic arrival models and/or the obliviousness of potential correlations in traffic flows. To alleviate this problem, Beck et al. [9] proposed to integrate statistical inference, based on past traffic data, into SNC. This important move opens the door for new opportunities to use the powerful analytical toolsets of SNC for real-world applications.

Along the same line of statistical modeling and inference, this chapter points out the potential benefits of using copula theory in SNC. With copula analysis and numerical experiments, we clearly show the region where copulas can be helpful and the best bound that SNC can possibly achieve. Statistical analysis on real-world trace data in an experiment with Skype conference calls shows that copula analysis can discover the (hidden) correlation between traffic flows, which in turn can help obtain tighter performance bounds. The discovery of copula modeling provides hints and also sheds light on the adaptive strategies in proprietary systems such as Skype. To the best of our knowledge, none of existing work has utilized copula analysis to enhance the capability of SNC in traffic modeling and performance bounds improvement.

3.2 Related Work

The theory of network calculus was first proposed by Cruz in 1991 [22, 23] for network performance evaluation. There are two main tracks of network calculus theory-deterministic network calculus (DNC) and stochastic network calculus (SNC). The details and the results of deterministic network calculus theory can be found in the books [51, 15]. This track of research only analyse the performance bounds of the worst case, which are too loose for practical use.

As an alternative, stochastic network calculus was developed. The basic properties and results of SNC are concluded in [43, 44]. It is generally non-trivial to derive tight performance bounds. Union bounds are generally used [44]. To achieve tighter performance bounds, independence case study is introduced in [44], which assumes the independence between arrivals and service. In addition, Martingales have been used to tighten the performance bounds [68, 20, 19, 42]. The basic idea is to construct a Martingale process and derive performance bounds with Doob's inequality.

Whether or not the above proposed assumptions accord well with the real traffic or service process is a problem. An error model may lead to failure when applying theoretical bounds on real case study. To avoid this situation, Beck *et al* [9] propose

statistical network calculus (StatNC). In his work, traffic models are established by measuring arrivals statistically. The performance bounds are also analysed in a statistical way. Due to advantage of StatNC, the performance bounds study reap great accuracy and robustness for different cases. Our work in Chapter 3 shares the same spirit as StatNC. Nonetheless, they are significantly different in that we make use of copula to capture the dependence between flows, while the work of [9] mainly focuses on the statistical estimation of a single flow.

3.3 Background of Stochastic Network Calculus

We introduce the notation and key concepts of stochastic network calculus [44, 54]. We assume that all arrival curves and service curves are non-negative and wide-sense increasing functions. Conventionally, $A(t)$ and $A^*(t)$ are used to denote the *cumulative* traffic that arrives and departs in time interval $(0,t]$, respectively, and $S(t)$ is used to denote the cumulative amount of service provided by the system in time interval $(0,t]$. For any $0 \leq s \leq t$, let $A(s,t) \equiv A(t) - A(s), A^*(s,t) \equiv A^*(t) - A^*(s)$, and $S(s,t) \equiv S(t) - S(s)$. By default, $A(0) = A^*(0) = S(0) = 0$.

We denote by $\mathcal{F}$ the set of non-negative wide-sense increasing functions, *i.e.*,

$$\mathcal{F} = \{f(\cdot) : \forall 0 \leq x \leq y, 0 \leq f(x) \leq f(y)\},$$

and by $\bar{\mathcal{F}}$ the set of non-negative wide-sense decreasing functions, *i.e.*,

$$\bar{\mathcal{F}} = \{f(\cdot) : \forall 0 \leq x \leq y, 0 \leq f(y) \leq f(x)\}.$$

For any random variable X, its distribution function, denoted by

$$F_X(x) \equiv Pr\{X \leq x\},$$

belongs to $\mathcal{F}$, and its complementary distribution function (or survival function), denoted by

$$\bar{F}_X(x) \equiv Pr\{X > x\},$$

belongs to $\bar{\mathcal{F}}$.

The $(\min, +)$ *convolution* of functions f and g is useful for SNC, and is defined

under the $(\min, +)$ algebra [15, 22, 51]:

$$(f \otimes g)(t) \equiv \inf_{0 \le s \le t} \{f(s) + g(t-s)\}. \tag{3.1}$$

In addition, the $(\min, +)$ *deconvolution* [15, 22, 51] of functions f and g is defined as:

$$(f \oslash g)(t) \equiv \sup_{s \ge 0} \{f(t+s) + g(s)\}. \tag{3.2}$$

For simplicity, we denote $[x]_1 \equiv min\{x, 1\}$ and $[x]^+ \equiv max\{x, 0\}$ in the following.

Stochastic traffic arrival curve and stochastic service curve are core concepts in stochastic network calculus, with the former used for traffic modeling and the latter for service modeling. In the literature, there are different definitions of stochastic arrival curve and stochastic service curve [44].

Definition 6. ***The*** *t.a.c.* ***model*** *[44]: A flow $A(t)$ is said to have a traffic-amount-centric (t.a.c.) stochastic arrival curve $\alpha \in \mathcal{F}$ with bounding function $f \in \bar{\mathcal{F}}$, denoted by*

$$A \sim_{tac} \langle f, \alpha \rangle,$$

if for all $t \ge s \ge 0$ and all $x \ge 0$,

$$Pr\{A(s,t) - \alpha(t-s) > x\} \le f(x). \tag{3.3}$$

In addition, we call $A(t-\delta, t), 0 \le \delta \le \Delta$ the ***statistic*** *of A within sliding window of size Δ.*

Intuitively, the above model means the cumulative amount of traffic arrivals in any time period is upper bounded by a function with some violation probability. The model is actually quite general and covers several broadly-used models. For example, the stochastically bounded burstiness (SBB) model [82] is a special case of the *t.a.c.* model by setting $\alpha(t-s) = \gamma \cdot (t-s)$. Following the same notation, when the traffic arrival $A(t)$ follows the SBB model with upper rate of γ and bounding function of f, we denote it as $A \sim_{SBB} \langle f, \gamma \rangle$.

Definition 7. ***The*** *v.b.c.* ***model*** *[44]: A flow $A(t)$ is said to have a virtual-backlog-centric (v.b.c.) stochastic arrival curve $\alpha \in \mathcal{F}$ with bounding function $f \in \bar{\mathcal{F}}$, denoted by*

$$A \sim_{vbc} \langle f, \alpha \rangle,$$

if for all $t \geq s \geq 0$ *and all* $x \geq 0$,

$$Pr\{\sup_{0\leq s\leq t}\{A(s,t)-\alpha(t-s)\}>x\}\leq f(x). \tag{3.4}$$

In addition, we call $\sup_{0\leq\delta\leq\Delta}\{A(t-\delta,t)\}$ *the* ***statistic*** *of* A *within sliding window of size* Δ.

Remark 1. *Assume that a (virtual) server with service rate* α *is fed with arrival* A. *The term* $\sup_{0\leq s\leq t}\{A(s,t)-\alpha(t-s)\}$ *represents the backlog of this virtual server at time* t. *Intuitively, the v.b.c model implies that the queue length of a virtual server of service rate* α *fed with the flow* A *is upper-bounded with some violation probability [44].*

We adopt the following model for services:

Definition 8. ***The*** *s.s.c.* ***model*** *[44]: A server is said to to provide a strict stochastic service curve (s.s.c.)* $\beta \in \mathcal{F}$ *with bounding function* $g \in \bar{\mathcal{F}}$, *denoted by*

$$S \sim_{ssc} \langle g, \beta \rangle ,$$

if during any period $(s,t]$ *the amount of service* $S(s,t)$ *provided by the server satisfies*

$$Pr\{S(s,t)<\beta(t-s)-x\}\leq g(x). \tag{3.5}$$

Remark 2. *In the literature, there are different definitions of stochastic arrival curve and stochastic service curve [44]. We adopt the above models with the consideration of their capability to model real-world traffic/services and the convenience to derive performance bounds.*

3.4 Insights of Copula Analysis

3.4.1 Basic Lemmas

In stochastic network calculus, we are often interested in the complementary distribution function of $Z = X+Y$, *i.e.*, $Pr\{Z > z\}$. The following two lemmas have been widely used in the derivation of stochastic bounds.

Lemma 1. ***General case*** *[44]: For the sum of two random variables* X *and* Y, $Z = X+Y$, *no matter whether* X *and* Y *are independent or not,* $\bar{F}_Z(z) \leq (\bar{F}_X \otimes \bar{F}_Y)(z)$.

Lemma 2. ***Independent case:*** *Assume that non-negative random variables X and Y are independent and $\bar{F}_X(x) \leq f(x)$ and $\bar{F}_Y(x) \leq g(x)$, where $f, g \in \bar{\mathcal{F}}$. Then, for all $x \geq 0$, $Pr\{X+Y > x\} \leq 1-(\bar{f} * \bar{g})(x)$, where $\bar{f}(x) = 1-[f(x)]_1, \bar{g}(x) = 1-[g(x)]_1$, and $*$ is the Stieltjes convolution operation.*

The following lemmas from copula analysis are useful for SNC:

Lemma 3. *Let Z be the sum of two random variables X and Y. The survival function of Z, $\bar{F}_Z(z)$ can be calculated in terms of $F_X(x)$, $F_Y(y)$ and their copula C_{XY}:*

$$\bar{F}_Z(z) = 1 - \iint_{x+y<z} \mathrm{d}C(F_X(x), F_Y(y)). \tag{3.6}$$

Proof. Let $f_Z(z)$ be the probability density function (pdf) of Z, and $F_Z(z)$ be its distribution function. Let $f_{XY}(x, y)$ be the joint probability density function of X and Y, and $F_{XY}(x, y)$ be the joint distribution function of X and Y. Since Z is the sum of X and Y, its pdf can be represented as

$$f_Z(z) = \int_{-\infty}^{\infty} f_{XY}(x, z-x)\mathrm{d}x = \int_{-\infty}^{\infty} f_{XY}(z-y, y)\mathrm{d}y. \tag{3.7}$$

Accordingly, $F_Z(z)$ is derived as follows:

$$\begin{aligned} F_Z(z) &= \int_{-\infty}^{z} f_z(t)\mathrm{d}t = \int_{-\infty}^{z}\int_{-\infty}^{\infty} f_{XY}(x, t-x)\mathrm{d}x\mathrm{d}t \\ &= \int_{-\infty}^{z-x}\int_{-\infty}^{\infty} f_{XY}(x, y)\mathrm{d}x\mathrm{d}y = \iint_{x+y<z} f_{XY}(x, y)\mathrm{d}x\mathrm{d}y \\ &= \iint_{x+y<z} \mathrm{d}F_{XY}(x, y) \\ &= \iint_{x+y<Z} \mathrm{d}C(F_X(x), F_Y(y)). \end{aligned} \tag{3.8}$$

With $\bar{F}_Z(z) = 1 - F_Z(z)$, Eq.(3.6) holds. □

Note that Lemma 3 can be extended to multivariate case.

Lemma 4. ***Copula case****: Let Z be the sum of two random variables X and Y. Then*

$$\hat{\bar{F}}_Z(z) \geq \bar{F}_Z(z) \geq \breve{\bar{F}}_Z(z), \tag{3.9}$$

where

$$\hat{\bar{F}}_Z(z) = 1 - \sup_{x+y=z} \{C_{lb}(F_X(x), F_Y(y))\}, \tag{3.10}$$

$$\breve{\bar{F}}_Z(z) = 1 - \inf_{x+y=z} \{\widetilde{C}_{lb}(F_X(x), F_Y(y))\}, \tag{3.11}$$

where C_{lb} is the Fréchet-Hoeffding lower bound copula defined in Theorem 3, and $\widetilde{C}_{lb}(u,v) = u + v - C_{lb}(u,v) = min(u+v, 1)$. The proofs of Lemma 4 are similar to those in [59] with slight modifications.

3.4.2 An Example of Copula Analysis

Markov modulated processes have been extensively used for representing multimedia traffic [78]. It has been shown that Markov modulated traffic could be captured with the stochastically bounded burstiness (SBB) model. Assume that we are given two Markov modulated processes $A_1 \sim_{SBB} \langle f_1, \gamma_1 \rangle$ and $A_2 \sim_{SBB} \langle f_2, \gamma_2 \rangle$. As a concrete example, we assume that both bounding functions, f_1 and f_2, have the exponential form [82] with mean values of r_1 and r_2, respectively. We are interested in modeling the superposition of A_1 and A_2, $A = A_1 + A_2$.

Let X and Y be exponentially distributed random variables with mean values of r_1 and r_2, respectively, and let $Z = X + Y$. With Lemma 1, we have the following bound, denoted as the *general bound* since it holds for any X and Y:

$$\bar{F}_Z(z) = \begin{cases} 1, & z < \eta \\ e^{-\frac{z-\eta}{r_1+r_2}}, & z \geq \eta \end{cases} \tag{3.12}$$

where $\eta = (r_1 + r_2)\ln(r_1 + r_2) - r_1 \ln(r_1) - r_2 \ln(r_2)$.

If we know that X and Y are independent, we have the following bound, denoted as the *independent bound*:

$$\bar{F}_Z(z) = \begin{cases} [(1 + \frac{z}{\gamma})e^{-\frac{z}{r}}]_1, & r_1 = r_2 = r \\ [\frac{r_1 e^{-\frac{z}{r_1}} - r_2 e^{-\frac{z}{r_2}}}{r_1 - r_2}]_1, & r_1 \neq r_2 \end{cases} \tag{3.13}$$

Based on Lemma 4, we have the following upper and lower bounds of $\bar{F}_Z$ from copula.

Theorem 5. *Let X and Y be exponentially distributed random variables with means r_1 and r_2, respectively. Let $\hat{\bar{F}}_Z$ and $\breve{\bar{F}}_Z$ be as in Lemma 4. Then*

$$\hat{\bar{F}}_Z(z) = \begin{cases} 1, & z < \eta \\ e^{-\frac{z-\eta}{r_1+r_2}}, & z \geq \eta \end{cases} \tag{3.14}$$

and

$$\breve{\bar{F}}_Z(z) = \begin{cases} 1, & z < 0 \\ e^{-\frac{z}{max(r_1,r_2)}}, & z \geq 0 \end{cases} \tag{3.15}$$

where $\eta = (r_1 + r_2)\ln(r_1 + r_2) - r_1\ln(r_1) - r_1\ln(r_2)$.

It is easy to show the following theorem to model the superposition of A_1 and A_2, $A = A_1 + A_2$.

Theorem 6. *Assume that $A_1 \sim_{SBB} \langle f_1, \gamma_1 \rangle$ and $A_2 \sim_{SBB} \langle f_2, \gamma_2 \rangle$, where f_1 and f_2 have the exponential form. The superposition of A_1 and A_2, $A \sim_{SBB} \langle g, \gamma_1 + \gamma_2 \rangle$, where g can be calculated:*

- *with Equation (3.12) (general bound, applicable to any situation),*
- *or with Equation (3.13) (independent bound, applicable when A_1 and A_2 are independent),*
- *or with Equation (3.14) (upper bound with copula), or with Equation (3.15) (lower bound with copula), or with Equation (3.6) (if the copula between A_1 and A_2 is known).*

Note that due to Lemma 4, the lower bound with copula indicates the tightest bound that we can possibly obtain with SNC when the upper rate is $\gamma_1 + \gamma_2$.

Figs. 3.1 and 3.2 show two numerical examples. We have the following interesting observations from the figures:

- The general bound is the same as the upper bound with copula, indicating that the general bound is actually the loosest bound.
- There is a clear gap between the independent bound and the lower bound with copula.

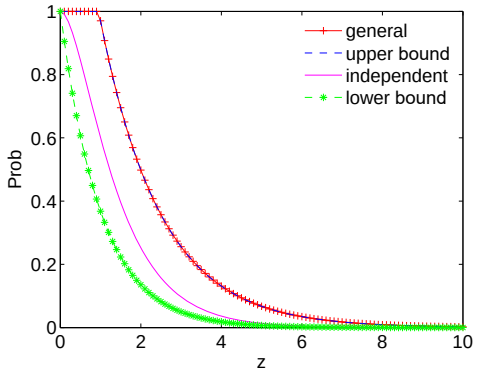


Figure 3.1: Different Bounds with $r_1 = 0.5$, $r_2 = 1$

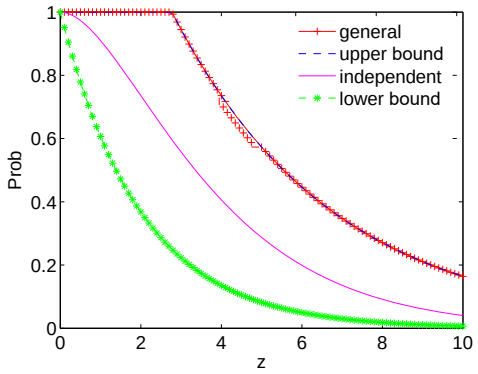


Figure 3.2: Different Bounds with $r_1 = 2$, $r_2 = 2$

Remark 3. *The gap between the independent bound and the lower bound with copula has important implication. When the dependence of flows is unclear or hard to determine, independent case analysis does not always lead to the best bound. There is much room for us to explore for improving stochastic bounds with copulas.*

3.4.3 Performance Bounds of SNC with Copulas

The following measures are of interest in service guarantee analysis:

- The backlog $\mathcal{B}(t)$ of flow A in the system at time t is defined as:

$$\mathcal{B}(t) = A(t) - A^*(t). \tag{3.16}$$

- The delay $\mathcal{D}(t)$ of flow A at time t is defined as:

$$\mathcal{D}(t) = \inf\{\tau \geq 0 : A(t) \leq A^*(t+\tau)\}. \tag{3.17}$$

If copula statistics are known, we have the following theorem to model the superposition of traffic flows:

Theorem 7. *Assume that $A_i \sim_{vbc} \langle f_i, \gamma_i \rangle (i = 1, \ldots, n)$. Assume that the statistic of $A_i (i = 1, \ldots n)$ within sliding window of size Δ, denoted as X_i, has a marginal distribution function F_{X_i}. Assume that the copula of $X_1, \ldots, X_n$, $C(F_{X_1}, \ldots, F_{X_n})$, is known. Based on the Definition 7 and the extended multivariate case of Lemma 3, the superposition of $A_1, \ldots, A_n$, $A \sim_{vbc} \langle g, \gamma_1 + \ldots + \gamma_n \rangle$, where g can be calculated as:*

$$g(z) = 1 - \int \cdots \int_{x_1 + \ldots + x_n < z} \mathrm{d}C(F_{X_1}(x_1), \ldots, F_{X_n}(x_n)). \tag{3.18}$$

With Theorem 7 in this thesis and Theorems 4.9 and 5.1 in [44], we have the following bound on backlog:

Theorem 8. ***Backlog Bound:*** *Consider a system with input flows $A_1, \ldots, A_n$. Assume that $A_i \sim_{vbc} \langle f_i, \gamma_i \rangle$. Assume that the statistic of $A_i (i = 1, \ldots n)$ within sliding window of size Δ, X_i, has a marginal distribution function F_{X_i} and that the copula of $X_1, \ldots, X_n$, $C(F_{X_1}, \ldots, F_{X_n})$, is known. Assume that the system provides to the input a service curve $S \sim_{ssc} \langle g, \beta \rangle$. The backlog $\mathcal{B}(t)$ is bounded by*

$$Pr\{\mathcal{B}(t) > x\} \leq (f \otimes g)(x - \alpha \oslash \beta(0)) \tag{3.19}$$

where

$$\alpha = \sum_{i=1}^{n} \gamma_i, \tag{3.20}$$

$$f = 1 - \int \cdots \int_{x_1 + \ldots + x_n < z} \mathrm{d}C(F_{X_1}(x_1), \ldots, F_{X_n}(x_n)). \tag{3.21}$$

In addition, with Theorem 7 in this thesis and Theorems 4.9 and 5.4 in [44], we have the following bound on delay:

Theorem 9. ***Delay Bound:*** *Consider a system with input flows $A_1, \ldots, A_n$. Assume that $A_i \sim_{vbc} \langle f_i, \gamma_i \rangle$. Assume that the statistic of $A_i (i = 1, \ldots n)$ within sliding*

window of size Δ, X_i, has a marginal distribution function F_{X_i} and that the copula of $X_1, \ldots, X_n$, $C(F_{X_1}, \ldots, F_{X_n})$, is known. Assume that the system provides to the input a service curve $S \sim_{ssc} \langle g, \beta \rangle$. The delay $\mathcal{D}(t)$ is bounded by

$$Pr\{\mathcal{D}(t) > h(\alpha + x, \beta)\} \leq (f \otimes g)(x) \tag{3.22}$$

where h denotes the maximum horizontal distance between two curves and

$$\alpha = \sum_{i=1}^{n} \gamma_i, \tag{3.23}$$

$$f = 1 - \int \cdots \int_{x_1 + \ldots + x_n < z} \mathrm{d}C(F_{X_1}(x_1), \ldots, F_{X_n}(x_n)). \tag{3.24}$$

Remark 4. *Following the same principle presented in [9], Theorems 8 and 9 both rely on the statistics of flow arrivals.*

3.5 Copula Modelling at Work

3.5.1 Copula Analysis in Real-world Applications

Real-world Experiments

To obtain an initial idea on traffic model of real-world flows and their dependence, we study traffic in Skype group calls as a preliminary step. The experiment scenario is shown in Fig. 3.3. Three users enter a Skype group call over a campus network. We name their IP addresses as IP 1, IP 2 and IP 3. During the group chatting, data flows (marked as dashed lines in Fig. 3.3) are transmitted between each pair of terminals. The outflows from IP 1 (marked as red dashed lines in Fig. 3.3) are identified as the 1st flow and the 2nd flow, respectively. The data packets of the two flows are captured with Wireshark. The captured information includes the frame number, the time, the source IP, the destination IP, the protocol, the length of packages, etc. To draw a reliable conclusion, we perform three independent experiments, each of which records traffic data of a group call for more than 20 minutes. Data collected from the three experiments is saved in Dataset 1, Dataset 2, and Dataset 3, respectively.

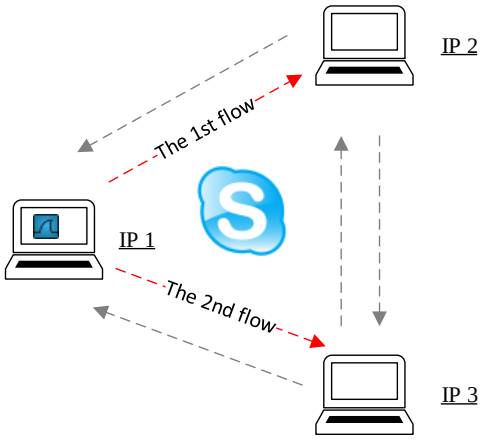


Figure 3.3: Experiment scenario

Traffic Modelling

We define a random variable **a** to represent the amount of data sent per unit of time (set as 1 second in our analysis). The values of observed samples of this random variable are denoted as a. Then the traffic during time interval $(s, t]$ can be regarded as the cumulative amount of traffic in each unit of time, *i.e.*

$$A(s,t) = \sum_{i=s+1}^{t} a^i, \tag{3.25}$$

where a^i is the observed value of **a** in i-th unit of time in the interval. Similarly, the traffic process $A(0,t)$ can be represented as a series of observed samples $a^1, a^2, \ldots, a^t$ of **a**. By modelling the distribution of random variable **a**, a traffic process becomes analytically easy to study.

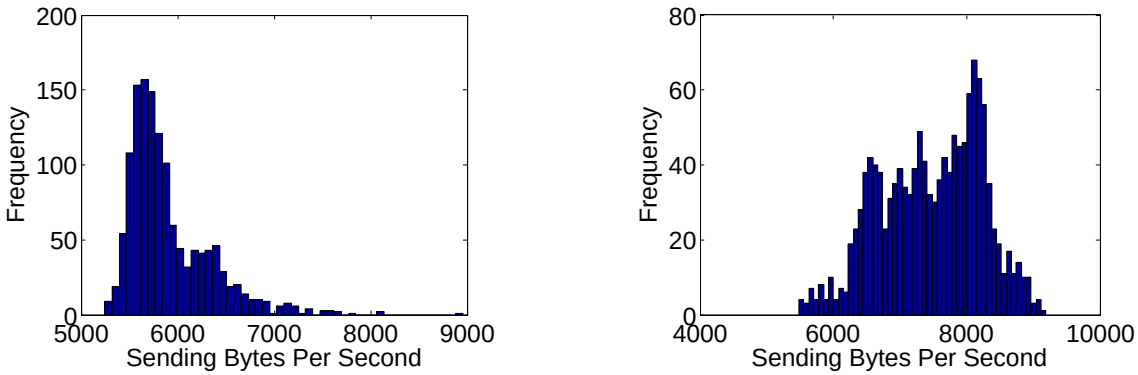
Therefore, we have two random variables to model in the datasets. One is the amount of data sent per unit of time in the 1st flow, denoted as $\mathbf{a}_1$; the other is the amount of traffic sent per unit of time in the 2nd flow, denoted as $\mathbf{a}_2$. To save space, we only show the histogram of sample values of $\mathbf{a}_1$ and $\mathbf{a}_2$ based on one dataset (Dataset 1), since the results from the other two datasets are similar. As shown in Fig. 3.4, the shape of the histograms seems to suggest a mixture of two Gaussian distributions[1]. The general form of cumulative distribution function (CDF) of the

[1]We also tested other distributions such as Gaussian and gamma distributions, but the data failed the test.

mixed distribution is

$$F(x) = \omega\Phi(\frac{x-\mu_1}{\sigma_1}) + (1-\omega)\Phi(\frac{x-\mu_2}{\sigma_2}), \tag{3.26}$$

where Φ is the CDF of the standard Gaussian distribution $Gaussian(0,1)$. The formula indicates the mixed distribution combining two weighted Gaussian distributions, $Gaussian(\mu_1, \sigma_1)$ and $Gaussian(\mu_2, \sigma_2)$. There are five parameters to estimate in Eq. (3.26), $\omega, \mu_1, \sigma_1, \mu_2, \sigma_2$. These parameters are computed by using the maximum likelihood estimate method on sample data.



(a) Histogram of samples of $\mathbf{a}_1$.

(b) Histogram of samples of $\mathbf{a}_2$.

Figure 3.4: Histogram of $\mathbf{a}_1$ and $\mathbf{a}_2$ based on samples in one dataset.

We then test the null hypothesis,

- the random variable $\mathbf{a}_i$ $(i = 1, 2)$ conforms to the mixture of two Gaussian distributions with parameters given by the parameter estimates.

The goodness of fit test is conducted with the Kolmogorov-Smirnov test [7]. The test results for the three datasets are shown in Table 3.1. The degrees of freedom are determined by the size of observed samples. From the table, the Kolmogorov-Smirnov statistic values D are always smaller than the critical values $D_{0.01}$. Therefore, the above null hypothesis cannot be rejected. Both $\mathbf{a}_1$ and $\mathbf{a}_2$ follow a mixture of two Gaussian distributions. This result suggests that Skype may adapt its sending rates along different channel conditions.

Copula-based Dependence between Flows

The dependence between the 1st flow and 2nd flow in each dataset is unknown to us. Copula analysis can help to disclose the hidden dependence structure. As described

Table 3.1: Kolmogorov-Smirnov goodness of fit test for $\mathbf{a}_1$ and $\mathbf{a}_2$ in three datasets.

		Dataset 1		Dataset 2		Dataset3	
Random variable		$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_1$	$\mathbf{a}_2$
Estimate of paramters	ω	0.60926	0.508744	0.574633	0.548374	0.434783	0.42081
	μ1	5674.983	6886.837	5617.563	6219.708	5760.396	5857.61
	μ2	6271.071	8072.268	6183.135	7427.538	6316.716	6463.853
	σ1	151.3938	532.1466	171.9808	388.4737	201.6061	183.0272
	σ2	470.381	394.1223	437.9055	605.1354	441.7937	461.2388
Statistical value D		0.033909	0.025224	0.023118	0.030246	0.031631	0.035658
Degree of freedom		1329		1837		1566	
Critical values $D_{0.01}$		0.0447		0.038		0.0412	

in the above section, traffic of each flow is represented by random variable $\mathbf{a}$. Then copula between random variables $\mathbf{a}_1$ and $\mathbf{a}_2$ shows how the two flows correlate with each other. In order to disclose the copula-based dependence between $\mathbf{a}_1$ and $\mathbf{a}_2$, we test three popular copulas, Gumbel copula, Frank copula and Clayton copula. By goodness-fit-test on these three copulas, we can quickly understand the dependence structure between $\mathbf{a}_1$ and $\mathbf{a}_2$.

The fitness to Gumbel, Frank and Clayton copulas is tested with "Blanket tests" based on empirical copula [36]. The main idea is to measure how far the empirical copula is from the tested copula. The test results are shown with P-value. Statistically, "the P-value can be viewed as a measure of fit, with larger values being better. This suggests that we could fit every distribution at our disposal, compute the test statistic for each fit, and then choose the distribution that yields the largest P-value" [7].

The fitness results for three copulas based on samples from the three datasets are listed in Table 3.2. Gumbel copula fits the samples best across all the three datasets. Therefore, it is suitable to use Gumbel copula to capture the dependence between flows $\mathbf{a}_1$ and $\mathbf{a}_2$.

Table 3.2: "Blanket" goodness of fit test for copula between $\mathbf{a}_1$ and $\mathbf{a}_2$ across three datasets.

		Dataset 1	Dataset 2	Dataset 3
Gumbel	θ	1.1464	1.0597	1.6791
	P-value	0.41	0.5	0.94
Frank	θ	1.2531	0.4483	4.465
	P-value	0.23	0.41	0.4
Clayton	θ	0.2057	0.0327	0.7574
	P-value	0.07	0.21	0

The Gumbel-based dependence actually reveals information about the transmission processes during Skype group calls in our experiments. It has been investigated that Skype adapts its sending rate to packet losses, packet delay, and available bandwidth [89]. Specifically, Skype will reduce the sending rate if the transmission channel is busy [89]. From the Gumbel-based dependence, we can infer that

- in most situations, the transmission channels for the two flows are not busy. Thus there is a relatively high probability that Skype arranges a high sending rate to the two destinations at the same time, causing the strong upper tail dependence;
- when the transmission channel for one flow becomes busy, Skype reduces the transmission rate to the corresponding destination, while the transmission rate to the other destination does not need to change, resulting in the weak lower tail dependence.

Remark 5. *Our test results do not exclude the possibility that the data may fit another possible distribution or another possible copula. Nevertheless, the completeness of statistical tests is not the main focus of our work, and our framework is generally applicable to other (possible) distributions and copulas.*

3.5.2 Copula Analysis with Simulated Traffic

Due to the difficulty in accurately tracking the buffer size allocated and used by Skype traffic, we study the performance bounds using simulated traffic. The simulated traffic flows follow the statistical model obtained in the above real-world experiments. On the service part, we simulate a constant rate server. To save space, we only show the backlog bounds. *The delay bounds can be studied with the similar method.*

Generation of Simulated Traffic

Consider a system with two input flows (A_1 and A_2) and a node with constant service rate to the input flows (R_1 to A_1 and R_2 to A_2). The two input flows follow the distributions and dependence structure same as those of two outflows in the above Skype group calls. In particular, the generated traffic amount per unit of time, $\mathbf{a}_1$ and $\mathbf{a}_2$, follow a mixture of two Gaussian distributions, and their correlation is modeled by Gumbel copula. Denote the CDF of $\mathbf{a}_1$ and $\mathbf{a}_2$ as $F_{\mathbf{a}_1}$ and $F_{\mathbf{a}_2}$, respectively,

and denote the copula between $\mathbf{a}_1$ and $\mathbf{a}_2$ as $C(F_{\mathbf{a}_1}, F_{\mathbf{a}_2}; \theta)$. The copula is chosen as Gumbel copula, and the parameter is chosen as $\theta = 1.6791$ according to our case study of real traffic Dataset 3. Note that the similar performance results can be obtained with other two datasets. Algorithm 1 shows the method to generate simulated traffic.

Algorithm 1 Traffic Generation Based On Given Distributions and Copula

Require: Distributions of $\mathbf{a}_1$ and $\mathbf{a}_2$, copula between them, the length of time of simulated process t

Ensure: Traffic data of two flows A_1 and A_2

1: **for** $i \leftarrow 1 : t$ **do**
2: Generate a random pair (u_1,u_2) based on given copula using the method introduced in [59];
3: Generate a sample of $\mathbf{a}_1$ within i-th unit of time by $a_1^i = F_{\mathbf{a}_1}^{-1}(u_1)$;
4: Generate a sample of $\mathbf{a}_2$ within i-th unit of time by $a_2^i = F_{\mathbf{a}_2}^{-1}(u_2)$;
5: **end for**
6: The sample sequence of $\mathbf{a}_1$, $\{a_1^1, a_1^2, \ldots, a_1^t\}$ represents traffic data of flow A_1;
7: The sample sequence of $\mathbf{a}_2$, $\{a_2^1, a_2^2, \ldots, a_2^t\}$ represents traffic data of flow A_2;

The output of Algorithm 1 is actually the traffic arrived in each unit of time of input flows A_1 and A_2. All the output traffic data is combined to be Simulated Dataset. According to output traffic, the traffic amount arrived within any time interval (s,t] can be computed in accumulative way with Eq. (3.27). Thus with simulated traffic data, the arrival process of two flows can be entirely recovered and further used for backlog bounds study.

$$A_1(s,t) = \sum_{i=s+1}^{t} a_1^i, \quad A_2(s,t) = \sum_{i=s+1}^{t} a_2^i. \tag{3.27}$$

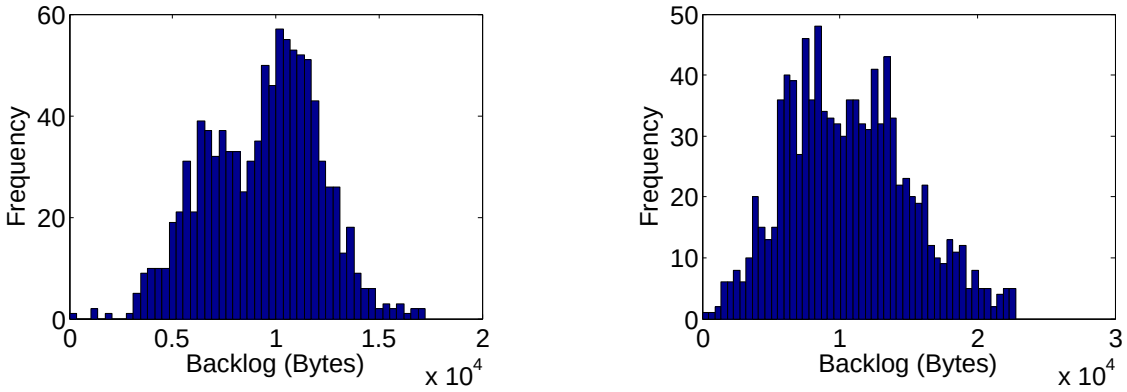
Backlog Bounds for Each Flow

Given the arrival $A_i(s,t)$ and a constant service rate R_i, the backlog $\mathcal{B}_i(t)$ is:

$$\mathcal{B}_i(t) = \sup_{0 \le s \le t} \{A_i(s,t) - R_i(t-s)\}, i = 1, 2. \tag{3.28}$$

If we characterize the backlog as a random variable $\mathbf{B}_i (i = 1, 2)$, the backlog sequence along time $\mathbf{B}_i(1)$, $\mathbf{B}_i(2)$, ..., $\mathbf{B}_i(t)$ are the observed samples of $\mathbf{B}_i$. Moreover, the backlog bounding function is the survival function of $\mathbf{B}_i$. Then the backlog bound can be estimated by the statistical distribution of backlog $\mathbf{B}_i$.

The service rate assigned to each flow equals its average arrival rate. By computation with Eq. (3.28), samples of $\mathbf{B}_i (i = 1, 2)$ can be obtained. The histograms of the sample values are shown in Fig. 3.5. The bimodal shaped histograms suggest that $\mathbf{B}_1$ and $\mathbf{B}_2$ may also follow a mixture of two Gaussian distributions, which is essentially inherited from the model of simulated input flows.



(a) Histogram of samples of $\mathbf{B}_1$.

(b) Histogram of samples of $\mathbf{B}_2$.

Figure 3.5: Histograms of $\mathbf{B}_1$ and $\mathbf{B}_2$ based on samples in simulated dataset.

The parameter estimation and Kolomogorov-Smirnov goodness of fit test results are shown in Table 3.3. For both $\mathbf{B}_1$ and $\mathbf{B}_2$, the statistical values are smaller than the critical values, indicating that they both follow a mixture of two Gaussian distributions. With the estimated parameters, the survival function of backlog variables $\mathbf{B}_1$ and $\mathbf{B}_2$ can be determined. Accordingly, the backlog bounds of flows A_1 and A_2 can be drawn, as shown in Fig. 3.6.

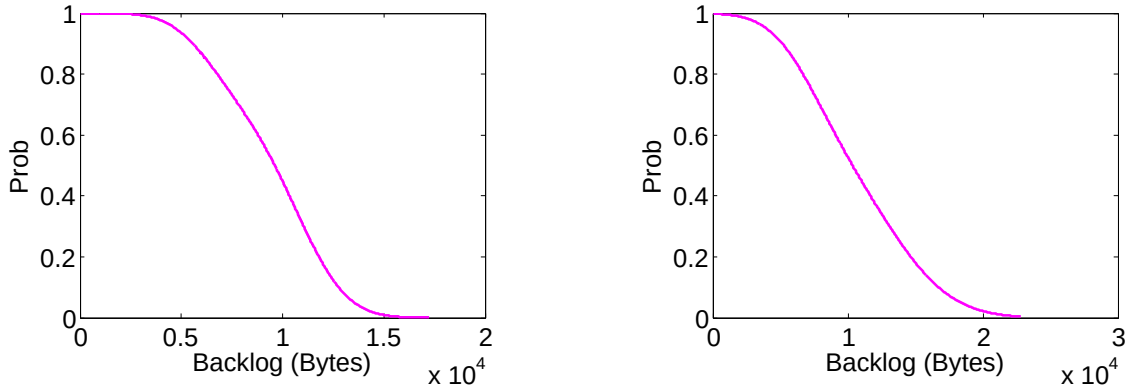
Remark 6. *The "raditional" way to obtain backlog bound with SNC is to derive the bound with traffic and service models. We treat the backlog as a random variable and models its statistical features directly. Nevertheless, this is not unusual. For the convenience of bound analysis, previous work [44] introduces some traffic models, such as the v.b.c. model, which could be considered as the same type of practice as ours, as per Definition 7 and Remark 1.*

Backlog Bound for Superposition of Two Flows

We next consider the backlog of the aggregated flow. The aggregated traffic is $A = A_1 + A_2$. The service rate assigned to A is $R_1 + R_2$. The backlog of A can be

Table 3.3: Kolmogorov-Smirnov goodness of fit test for backlog based on simulated dataset

Random variable		$\mathbf{B}_1$	$\mathbf{B}_2$
Estimate of paramters	ω	0.316657	0.31119
	$\mu 1$	6402.2	6912.625
	$\mu 2$	10741.37	12382.1
	$\sigma 1$	1650.444	2439.608
	$\sigma 2$	1930.165	4116.222
Statistical value D		0.021	0.0233
Degree of freedom		1000	
Critical values $D_{0.01}$		0.0515	



(a) Backlog bound curve of flow A_1.

(b) Backlog bound curve of flow A_2.

Figure 3.6: Backlog bound curves of two input flows of the simulated system.

represented as the summation of backlogs of A_1 and A_2:

$$\begin{aligned}\mathcal{B}(t) &= \sup_{0\le s\le t}\{A(s,t)-R(t-s)\},\\ &\le \sup_{0\le s\le t}\{A_1(s,t)-R_1(t-s)\}+\sup_{0\le s\le t}\{A_2(s,t)-R_2(t-s)\},\\ &=\mathcal{B}_1(t)+\mathcal{B}_2(t).\end{aligned} \tag{3.29}$$

Based on the analysis in the previous section, we can obtain the survival functions of $\mathbf{B}_1$ and $\mathbf{B}_2$, denoted as $\bar{F}_{\mathbf{B}_1}$ and $\bar{F}_{\mathbf{B}_2}$, respectively. With Lemma 1, the general bound of $\mathcal{B}(t)$ can be calculated as:

$$Pr\{\mathcal{B}(t)>x\}\le(\bar{F}_{\mathbf{B}_1}\otimes\bar{F}_{\mathbf{B}_2})(x). \tag{3.30}$$

By introducing a proper copula capturing the correlation between $\mathbf{B}_1$ and $\mathbf{B}_2$, the

backlog bound of $\mathcal{B}(t)$ can be calculated with Lemma 3 and is tighter:

$$Pr\{\mathcal{B}(t) > x\} \leq 1 - \int\int_{b_1+b_2<x} dC(F_{\mathbf{B}_1}, F_{\mathbf{B}_2}). \tag{3.31}$$

To identify a proper copula between $\mathbf{B}_1$ and $\mathbf{B}_2$, we do the fitness test based on Gumbel, Frank and Clayton copulas, respectively. The estimated parameters and the test results are shown in Table 3.4. Clearly, Clayton is the one that best models the dependence between $\mathbf{B}_1$ and $\mathbf{B}_2$. The fitness of Clayton copula shows that the backlogs of two flows are more lower tail dependent. That is, the probability that backlogs of small size appear in both traffic flows at the same time is higher.

Table 3.4: "Blanket" goodness of fit test for copula between $\mathbf{B}_1$ and $\mathbf{B}_2$ based on simulated dataset

Gumbel	θ	2.48
	P -value	0.03
Frank	θ	9.3526
	P -value	0.21
Clayton	θ	3.5
	P -value	0.68

Given the Clayton copula with the estimated parameter and the known marginal distributions $F_{\mathbf{B}_1}$ and $F_{\mathbf{B}_2}$, the copula-based backlog bound can be computed with Eq. (3.31). Note that the copula-based bound obtained here is a special case of Theorem 8, for the service process is simplified as a constant-rate service. Both the general bound and the copula-based bound are shown in Fig. 3.7. We also label the values $x_{bound} = \inf_x Pr(\mathcal{B}(t) > x) \leq 0.1$ from simulation, copula bound, and general bound with vertical lines in the figure. Practically, the values x_{bound} bound backlog with a small violation probability (less than 0.1). The value from copula bound is very close to the simulation result and much smaller than the value from the general bound. It is clear that the copula-based bound is closer to reality and tighter than the general one.

3.6 Summary

Integrating the statistical method in SNC has been shown to be promising [9]. With a concrete real-world case study and numerical examples, this chapter illustrates the benefit of applying copula analysis in SNC for tighter performance bounds. This

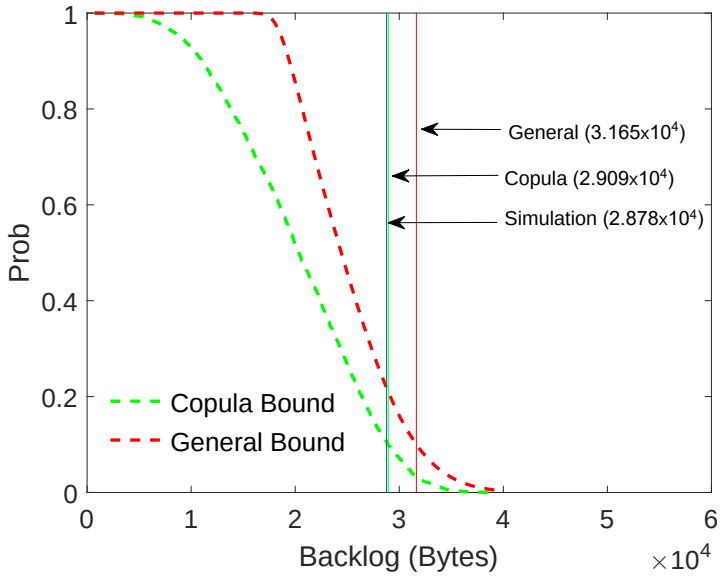


Figure 3.7: Backlog bound for aggregate traffic A.

analysis also sheds light on several important issues in SNC, such as the region where we can take advantage of dependence of random processes, and the tightest bound that SNC can possibly achieve.

Chapter 4

Copula Analysis of Temporal Dependence of Markov Modulated Poisson Process

4.1 Introduction

Markov modulated Poisson process (MMPP) is the doubly stochastic Poisson process whose arrival rate is modulated by an irreducible continuous time Markov chain (CTMC) independent with the arrival process [31]. Specifically, the arrival process is a Poisson process with arrival rate λ_j whenever the CTMC is in state j. MMPP was first proposed by Yechiali and Naor to model non-homogeneous Poisson arrival process in queueing systems [86]. Compared with traditional Poisson process, MMPP allows the arrival rates to vary from time to time, making the model more flexible. Besides, MMPP is effective to capture burst arrivals and sudden changes in arrivals since it can integrate significantly different rates into one model. All these benefits make MMPP a widely applied model for the arrival processes in networks [17, 38], for the processes that show pattern changes [24], and for burst events detection [79, 41].

The good properties and the broad applications of MMPP are all on the basis of the temporal dependence carried by MMPP. Essentially, the dependence/correlation among inter-arrival times is the main difference between MMPP and Poisson process [10]. In the model of MMPP, the inter-arrival times are not independent. The dependence between inter-arrival times comes from CTMC that modulates the state switches over time. With the dependence structure of MMPP, we can better under-

stand the process and predict its trend. For example, when we model or detect traffic of networks with MMPP, the temporal dependence of MMPP can be the objective to match with that of the real traffic trace. Another example is to model web traffic or traffic in cloud with MMPP [70, 69, 64]. In this case, resource provisioning based on MMPP arrivals is the problem of interest. The capability of predicting arrivals based on temporal dependence structures is critical in designing the resource provisioning policy.

Existing theoretical studies of MMPP mainly fall into two categories. One track of studies is to use MMPP as the input of the queueing system and study the queueing performance. Current representative works include [17, 70]. The other track of studies is to develop algorithms to estimate the parameters of MMPP. Recent developments cover the algorithms of fitting MMPP to IP traffic traces [6], the expectation-maximization (EM) based algorithms to learn MMPP as a type of Markovian Arrival Process [63], the algorithms to learn MMPP through the detection of change points along with the arrival rates estimation [14], and the online learning algorithms by modeling MMPP as a Hidden Markov Model [16]. All these learning algorithms are either based on the arrival times, or the number of arrivals within every unit of time (arrival counts). Despite the abundant existing theoretical results on MMPP, there still is a large gap in the formal analysis of the dependence structure of MMPP in the literature. This gap is reflected in the following aspects.

First, the temporal dependence of MMPP is not well understood. The existing results related to the MMPP temporal dependence are all on the basis of covariance/autocorrelation. Neuts derived covariance between arrival counts over any two time slots for stationary MMPP in 1989 [61], which is still the strongest result known. This covariance result is not sufficient for many applications. To begin with, the covariance is not easy to compute due to the matrix exponential and matrix inverse involved (especially when the number of states becomes large). For 2-state stationary MMPP, the closed-form of the covariance between arrival counts was given by Andersen and Nielsen in 1998 [3]. For multi-state MMPPs, their covariances are usually obtained approximately by statistical counting on simulated traces [64, 16, 3]. Furthermore, the covariance or the autocorrelation is only capable of measuring the *linear* dependence degree over time. However, the MMPP network traffic may contain temporal dependence more complex than linear dependence. Through a detailed example given in Section 4.3.2, it is clear that the covariance only captures MMPP dependence structure partially and is far from reflecting its whole dependence struc-

ture. This motivates us to search for the exact and functional temporal dependence structure of MMPPs.

Second, there is no analysis on the temporal dependence in the superposition of MMPPs, *i.e.*, the aggregation of multiple flows, each modeled as an MMPP [39]. Although it has been proved that the superposition of MMPP is still an MMPP [31], analysing the superposition of MMPP becomes intractable in real applications due to the exponential increase of the number of states. For instance, the superposition of two 20-states MMPP is computationally expensive to solve [39]. In other words, simply treating the superposition of MMPP as one MMPP of higher number of states would not work in practice. The temporal dependence of superposition of MMPPs thus requires a different analytical method.

Copula, an advanced dependence measure that links marginals into joint distributions, is ideal for modeling the temporal dependence of MMPP. First, copula can be constructed theoretically based on the analysis on marginals and joint distributions of the observations in MMPP. Second, copula is capable of capturing all the characteristics from dependence structures. Beyond the linear dependence, it characterizes functional dependence structure and carries abundant dependence information. Third, with the help of copula, it is easy to avoid the explosion of the number of parameters when modeling the superposition of MMPPs, and it is computationally tractable to calculate the temporal dependence of superposed MMPPs. Finally, the invariant property of copula keeps the dependence measure stable even when MMPP trace changes functionally. In this chapter, we build the theoretical copula to capture the temporal dependence of both single and superposed MMPPs.

4.2 Related Work

The Markov Modulated Poisson Process (MMPP) was first applied in the network domain in 1971 [86]. Since then, tremendous research efforts have been devoted to MMPP. Early theoretical results and applications of MMPP were outlined in the review [31] and references therein. In brief, the review includes the theoretical results of the characterization of MMPP, the statistical moments of MMPP arrivals, and the superposition of independent MMPPs. Afterwards, MMPP was further studied as the arrival input of queueing systems. Furthermore, various learning algorithms for parameter estimation of MMPP were proposed.

Among the literature of MMPP, the research that related to temporal dependence

modeling is summarized as follows. MMPP and other Markovian arrival processes were generalized into the versatile Markov point process in [61]. The covariance between arrival counts was derived for the versatile Markov point process. The closed form of the covariance of 2-state MMPP was given in [3]. The covariance was further derived into an asymptotic form and used for learning parameters. In [64, 16, 3], covariance was the evaluation metric for goodness of fitting test for MMPP, and it was computed empirically from simulated trace of fitted MMPP. Different from the above works, our work in Chapter 4 derives the theoretical results on temporal dependence of MMPP in terms of copula, which represents functional dependence.

In the case of superposition of MMPPs, its mathematical form has been given in [31], but the parameter computation of superposed MMPPs is complex due to the explosion of state number. To reduce the computational complexity, recent efforts have been made to reduce the number of states and obtain an approximate solution [39, 88]. Our work in Chapter 4 focuses on the exact and tractable solution of temporal dependence in the superposition of MMPPs.

4.3 Preliminaries

4.3.1 Markov Modulated Poisson Process

We introduce the definition and key concepts of MMPP.

Definition 9. *A **Markov-modulated Poisson Process** (MMPP) [31] is constructed by varying the arrival rate of a Poisson process according to an m-state irreducible continuous-time Markov chain (CTMC). In particular, when the Markov Chain is in state j, the arrivals follow a Poisson process of rate λ_j. Therefore, an MMPP can be parameterized by the Q matrix [73] of CTMC and the m Poisson arrival rates, $\Lambda = (\lambda_1, \ldots, \lambda_m)$.*

We thus denote an MMPP by parameters (Q, Λ).

Definition 10. ***Environment-stationarity** of an MMPP [31]: An MMPP (Q, Λ) is considered to be environment-stationary if its associated CTMC is stationary.*

For an environment-stationary MMPP, the stationary distribution of the states, $\Pi = (\pi_1, \ldots, \pi_m)$, is determined by solving the equation $\Pi Q = 0$. In our analysis of MMPP, we **only consider the environment-stationary MMPP.**

Since the superposition of MMPPs is still an MMPP [31], to distinguish regular MMPP with superposition of MMPPs, in this thesis either the term **single MMPP** or **MMPP** refer to an MMPP not created from superposition. We introduce the following terms to refer the superposition of MMPPs:

Definition 11. ***Superposition of independent homogeneous MMPPs****: An MMPP is called* ***HoMMPP*** *if it is a superposition of multiple independent homogeneous MMPPs. All constituent MMPPs have the same parameter* (Q, Λ)*.*

Definition 12. ***Superposition of independent heterogeneous MMPPs****: An MMPP is called* ***HeMMPP*** *if it is a superposition of multiple independent heterogeneous MMPPs. The constituent MMPPs carry different parameters* $({}_1Q, {}_1\Lambda), ({}_2Q, {}_2\Lambda)$*, ...,* $({}_lQ, {}_l\Lambda)$*..., where* $({}_lQ, {}_l\Lambda)$ *denotes the parameters of the* l*-th constituent MMPP.*

Definition 13. ***Arrival counts*** *of MMPP are a sequence of random variables representing the number of arrivals in disjoint equal-sized small time intervals, called time slots. Denote the sequence of time slots as* $I_1, I_2, \ldots, I_n$*, and the random variable representing the arrival count of single MMPP in* I_i *as* A_i*, of superposition of* l *independent MMPPs in* I_i *as* A_i^l*.*

Remark 7. We denote the length of each time slot as Δ. For MMPP modeling we assume Δ is short enough such that the state transition of MMPP within one time slot is negligible. To keep this assumption valid, we recommend that the length of time slot be no larger than the smallest average time of MMPP staying on one state, *i.e.*, $\Delta \leq \frac{1}{\max_j |q_{jj}|}$ where q_{jj} is the diagonal element in the j-th row of matrix Q. Under this condition, the number of state transitions in one time slot can be ignored and the arrival rate in one time slot is (approximately) stable. Experiments in [63] have showed that the parameter estimation based on arrival counts becomes inaccurate when $\Delta > \frac{1}{\max_j |q_{jj}|}$, indicating that a large value of Δ would make the arrival counts lack enough information to retrieve the MMPP. In other words, we assume that the state transitions occur only at the boundaries of time slots. This approximation has been used in previous research, *e.g.*, in [63].

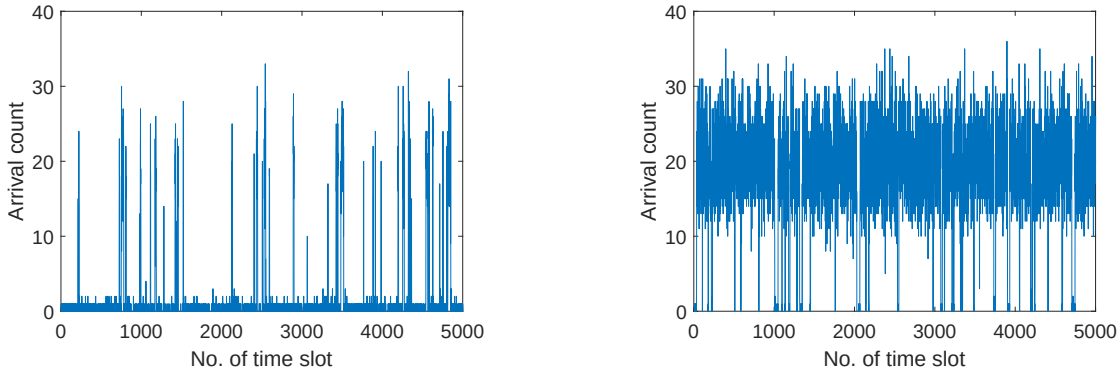
4.3.2 Why Do Existing Results Not Suffice?

The strongest result so far that discloses the temporal dependence of MMPP is from [61], where the covariance or the autocorrelation of arrival counts over different

time slots is given. Covariance, however, is only capable of capturing linear dependence. MMPP trace may contain temporal dependence much more complex than linear dependence. To illustrate the pitfalls of covariance, we consider two MMPPs with their parameters as $({}_1Q, {}_1\Lambda)$ and $({}_2Q, {}_2\Lambda)$ shown below:

$${}_1Q = \begin{pmatrix} -0.1 & 0.1 \\ 1 & -1 \end{pmatrix}, \quad {}_1\Lambda = (2, 200); \qquad {}_2Q = \begin{pmatrix} -0.1 & 0.1 \\ 1 & -1 \end{pmatrix}, \quad {}_2\Lambda = (200, 2).$$

We simulate traces from these two MMPPs: Trace 1 is from MMPP $({}_1Q, {}_1\Lambda)$; Trace 2 is from MMPP $({}_2Q, {}_2\Lambda)$. The traces are analysed by arrival counts. Specifically, the number of arrivals in i-th timeslot is denoted as $A_i (i \in \mathbb{N})$. The arrival counts of the two traces are shown in Fig. 4.1. From the figure, the traces from the two MMPPs are very different. To study their dependence, we first analyse the covariances of two MMPPs and then visualize their temporal dependence by the joint distribution of successive arrival counts.



(a) Trace 1 (b) Trace 2

Figure 4.1: Arrival counts of the two traces

The theoretical form of the covariance of a 2-state MMPP is given by Eq.(3) in Section II of [3]. Based on the given covariance function, the covariances between A_i and $A_{i+i'} (i' \in \mathbb{N},$ is the time lag) of the two MMPPs in this example is theoretically the same. In Fig. 4.2, we use the green plots to show that the theoretical covariances of the two MMPPs (from the theoretical analysis with Eq.(3) of [3]) are all the same over different time lags. We also plot the empirical covariances calculated from the simulated traces. The covariances of two traces are close, though they vary slightly from the theoretical results. So in terms of covariance, the two MMPPs show the same dependence structure.

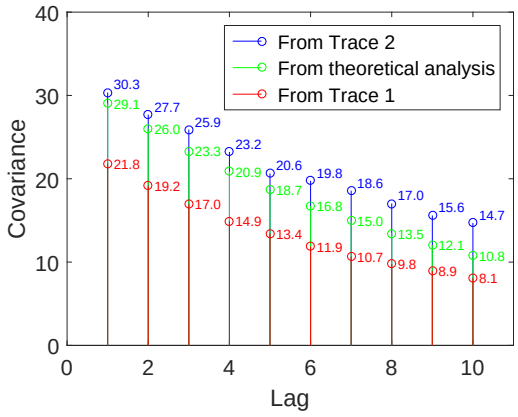
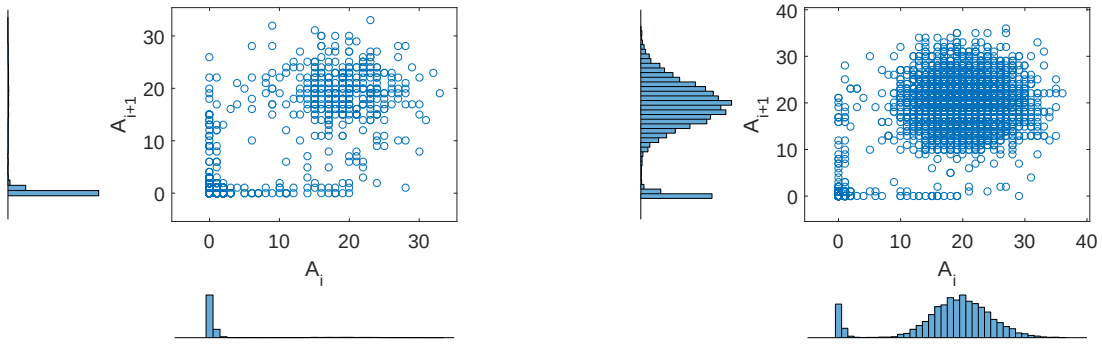


Figure 4.2: Covariances of two MMPPs over different time lags

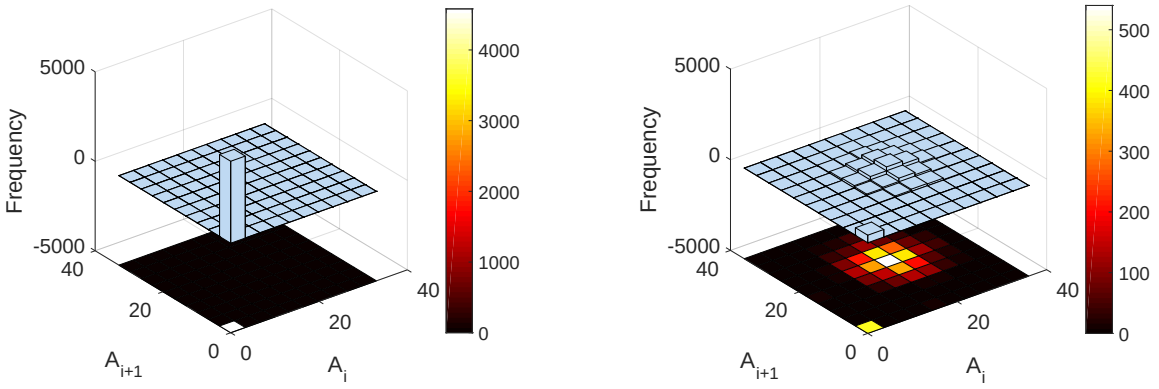
To obtain the full view of the dependence structure, we visualize the joint behaviour of A_i and A_{i+1} by the scatter plots with marginal histograms in Fig. 4.3 and their bivariate frequency histograms with heat map in Fig. 4.4. From the two figures, we can observe that the joint behaviour of two successive arrival counts is quite different in the two MMPPs. Therefore, it is clear that the two MMPPs have different temporal dependence between A_i and A_{i+1}.



(a) Trace 1 (b) Trace 2

Figure 4.3: Scatter plot with marginal histograms of A_i and A_{i+1} in two traces

In the above simple example, the two MMPPs have the same covariance theoretically. However they generate traces with significantly different temporal dependence structures. Therefore, covariance, only measuring partial information from dependence, is not sufficient to represent MMPP dependence. This motivates us to seek a better dependence structure to characterize temporal dependence beyond linear



(a) Trace 1 (b) Trace 2

Figure 4.4: Bivariate frequency histogram (upper layer) with its heat map (lower layer)

scope when modeling network traffic with MMPP. We tackle this challenge with copula analysis. We will apply both theoretical way (Section 4.4) and parametric copula modeling (Section 4.5) to construct the copula of MMPP and superposed MMPP.

4.4 Theoretical Copula Analysis for MMPP, HoMMPP and HeMMPP

4.4.1 Theoretical Copula Analysis for Single MMPP

We first study an m-state MMPP with parameters (Q, Λ). Based on Definition 13 and Remark 7, the state in I_i is considered as stable, thus defined as a random variable S_i. Denote the transition matrix by $P(t) = [p_{j_1 j_2}(t)]$, where $p_{j_1 j_2}(t)$ is the probability that the CTMC switches from state j_1 to state j_2 after time t. $P(t) = e^{Qt}$ can be calculated with numerical methods such as those introduced in Chapter 6.8 of [73]. As Δ is small, $P(\Delta)$ relates to Q matrix in the following way:

$$\begin{aligned} p_{j_1 j_2}(\Delta) &= 1 + q_{j_1 j_2}\Delta + o(\Delta), \quad j_1 = j_2; \\ p_{j_1 j_2}(\Delta) &= q_{j_1 j_2}\Delta + o(\Delta), \quad j_1 \neq j_2 \end{aligned} \tag{4.1}$$

where $o(\Delta)$ is an infinitesimal. Therefore, by a simple calculation, $P(\Delta)$ is approximately equal to matrix $Q\Delta$ plus an identity matrix.

The MMPP traffic will be analysed in terms of arrival counts A_i, and the temporal

dependence of MMPP will be the dependence between A_i and $A_{i+i'}$ with time lag as i'. Under environmental-stationarity, the arrival counts of all time slots (*i.e.*, A_i with any i) share the same marginal distribution function, denoted as M. Similarly, the copula between A_i and $A_{i+i'}$ will be a function invariant on time slot label i but only variant along time lag i', thus is denoted as $C_{i'}$. In the following, we derive the marginal distribution function M in Theorem 10 and the copula $C_{i'}$ in Theorem 11.

Theorem 10. *Let x_i be the sample value of A_i, the marginal distribution of A_i on x_i is*

$$M(x_i) \equiv Pr(A_i \leq x_i) = \sum_{j=1}^{m} \pi_j G_j(x_i) = \Pi \mathbb{G}(x_i) \tag{4.2}$$

where

- $G_j(x_i) \equiv Pr(A_i \leq x_i | S_i = j) = e^{-\lambda_j \Delta} \sum_{k=0}^{k=x_i} \frac{(\lambda_j \Delta)^k}{k!}$,
- $\mathbb{G}(x_i) \equiv [G_1(x_i), \cdots, G_m(x_i)]$ *is a conditional marginal vector.*

Proof.

$$M(x_i) \equiv Pr(A_i \leq x_i) = \sum_{j=1}^{m} Pr(A_i \leq x_i | S_i = j) Pr(S_i = j) = \sum_{j=1}^{m} \pi_j G_j(x_i) = \Pi \mathbb{G}(x_i).$$

□

Theorem 11. ***(Single MMPP copula)*** *Let $u_i \equiv M(x_i)$, the copula of any two arrival counts, A_i and $A_{i+i'}$ ($i' \in \mathbb{N}$), can be calculated as:*

$$C_{i'}(u_i, u_{i+i'}) = \mathbb{G}(M^{-1}(u_i)) diag(\Pi) P(i'\Delta) \mathbb{G}(M^{-1}(u_{i+i'}))^T, \tag{4.3}$$

where

- M^{-1} *is the inverse function of M defined by (4.2),*
- $diag(\Pi)$ *is a square diagonal matrix with the elements of vector Π on the main diagonal,*
- $\mathbb{G}(M^{-1}(v))^T$ *is the transpose of $\mathbb{G}(M^{-1}(v))$.*

Proof. The joint distribution of A_i and A_{i+1} is derived as

$$
\begin{aligned}
&F_{A_iA_{i+i'}}(x_i, x_{i+i'}) \equiv Pr(A_i \le x_i, A_{i+i'} \le x_{i+i'}) \\
=&\sum_{j_2=0}^{m}\sum_{j_1=0}^{m} Pr(A_i \le x_i, A_{i+i'} \le x_{i+i'} | S_i = j_1, S_{i+i'} = j_2) Pr(S_i = j_1, S_{i+i'} = j_2) \\
=&\sum_{j_2=0}^{m}\sum_{j_1=0}^{m} Pr(A_i \le x_i | S_i = j_1) Pr(A_{i+1} \le x_{i+i'} | S_{i+1} = j_2) Pr(S_i = j_1) Pr(S_{i+i'} = j_2 | S_i = j_1) \\
=&\sum_{j_2=0}^{m}\sum_{j_1=0}^{m} G_{j_2}(x_{i+i'}) p_{j_1 j_2}(i'\Delta) G_{j_1}(x_i) \pi_{j_1} \\
=&\mathbb{G}(x_i) diag(\Pi) P(i'\Delta) \mathbb{G}(x_{i+i'})^T.
\end{aligned}
$$

With the inverse method based on Theorem 4, the copula between A_i and $A_{i+i'}$ is constructed as

$$
\begin{aligned}
C_{i'}(u_i, u_{i+i'}) &= F_{A_iA_{i+i'}}(M^{-1}(u_i), M^{-1}(u_{i+i'})) \\
&= \mathbb{G}(M^{-1}(u_i)) diag(\Pi) P(i'\Delta) \mathbb{G}(M^{-1}(u_{i+i'}))^T.
\end{aligned}
$$

□

The copula $C_{i'}$ in Theorem 11 is the theoretical copula that models the dependence of two arrival counts in MMPP. The number of time slots between two arrival counts are specified by the value of i'. We name this theoretical copula as *single MMPP copula*.

4.4.2 Theoretical Copula Analysis for HoMMPP

For a better understanding, we start from HoMMPP to explore the copula of superposition of MMPPs. HoMMPP is a good model for many network system, such as Internet core routers, where the incoming traffic may be a superposition of multiple independent homogeneous MMPP traffic traces. We consider a HoMMPP with the number of constituent MMPPs as $l (l \in \mathbb{N})$, each of which has the same parameters (Q, Λ). **All notations related to the HoMMPP is numbered with l on their top right.** That is, the HoMMPP has arrival counts random variable as A_i^l, marginal distribution of A_i^l as M^l, copula between A_i^l and $A_{i+i'}^l$ as $C_{i'}^l$, *etc.* Note that when $l = 1$, the HoMMPP regress to single MMPP, the notations can omit l to be consistent with those defined in Section 4.4.1.

For HoMMPP, it is hard to derive the theoretical copula directly. When l is getting large, number of states of CTMC associated to HoMMPP will explode, and the joint distribution of A_i^l and $A_{i+i'}^l$ can hardly expressed in closed-form. To tackle this difficulty, we derive following theorems, which are helpful to reveal HoMMPP copula. These two theorems will be the basis of an algorithmic approach to compute HoMMPP copula introduced later in Section 4.4.4.

Theorem 12. *Let x_i denote the sample value of A_i^l, the marginal of A_i^l is*

$$M^l(x_i) = \sum_{l_1+\cdots+l_m=l} \binom{l_1}{l} \binom{l_2}{l-l_1} \cdots \binom{l_m}{l-l_1-\cdots-l_{m-1}} * \pi_1^{l_1}\pi_2^{l_2}\cdots\pi_m^{l_m} * Po((l_1\lambda_1+\cdots+l_m\lambda_m)\Delta, x_i), \tag{4.4}$$

where $\binom{k}{n}$ is the combinatorial number of choosing k from n, and $Po(\lambda, x_i)$ represents the Poisson cumulative distribution of value x_i, with parameter λ.

Proof. Assume the number of MMPP in State j in time slot I_i is l_j $(j = 1, 2, \ldots, m)$. The probability that the HoMMPP is at the above allocation of states is $\binom{l_1}{l} \binom{l_2}{l-l_1} \cdots \binom{l_m}{l-l_1-\cdots-l_{m-1}} * \pi_1^{l_1}\pi_2^{l_2}\cdots\pi_m^{l_m}$. Since the superposition of two Poisson processes with rate λ_1 and λ_2 is a Poisson process with rate $\lambda_1+\lambda_2$, under the assumed state combination A_i^l follows Poisson distribution with parameter of $(l_1\lambda_1+\cdots+l_m\lambda_m)\Delta$. Adding up all the possible allocations of states leads to the marginal form in Theorem 12. □

We define the copula gradients here for further analysis of HoMMPP copula.

Definition 14. ***Single MMPP copula gradient*** *$\nabla C_{i'}$ is defined as*

$$\begin{aligned} &\nabla C_{i'}(u_i, u_{i+i'}) = \nabla C_{i'}(M(x_i), M(x_{i+i'})) \\ \equiv &C_{i'}(M(x_i), M(x_{i+i'})) + C_{i'}(M(x_i-1), M(x_{i+i'}-1)) \\ &- C_{i'}(M(x_i), M(x_{i+i'}-1)) - C_{i'}(M(x_i-1), M(x_{i+i'})); \end{aligned} \tag{4.5}$$

HoMMPP/HeMMPP copula gradient $\nabla C^l_{i'}$ *is defined as*

$$\begin{aligned}
&\nabla C^l_{i'}(u_i, u_{i+i'}) = \nabla C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) \\
\equiv &C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) + C^l_{i'}(M^l(x_i - 1), M^l(x_{i+i'} - 1)) \\
&- C^l_{i'}(M^l(x_i), M^l(x_{i+i'} - 1)) - C^l_{i'}(M^l(x_i - 1), M^l(x_{i+i'})).
\end{aligned} \tag{4.6}$$

Lemma 5. *Single MMPP copula gradient can be simply regarded as* $\nabla C_{i'}(M(x_i), M(x_{i+i'})) = Pr(A_i = x_i, A_{i+i'} = x_{i+i'})$*; Similarly,* $\nabla C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) = Pr(A^l_i = x_i, A^l_{i+i'} = x_{i+i'})$.

Proof. Based on the definition of single MMPP copula gradient and the fact that the arrival counts follow discrete marginal distributions, we have

$$\begin{aligned}
&\nabla C_{i'}(M(x_i), M(x_{i+i'}) \\
=&C_{i'}(M(x_i), M(x_{i+i'})) + C_{i'}(M(x_i - 1), M(x_{i+i'} - 1)) \\
&- C_{i'}(M(x_i), M(x_{i+i'} - 1)) - C_{i'}(M(x_i - 1), M(x_{i+i'})) \\
=&Pr(A_i \le x_i, A_{i+i'} \le x_{i+i'}) + Pr(A_i \le x_i - 1, A_{i+i'} \le x_{i+i'} - 1) \\
&- Pr(A_i \le x_i, A_{i+i'} \le x_{i+i'} - 1) - Pr(A_i \le x_i - 1, A_{i+i'} \le x_{i+i'}) \\
=&Pr(A_i = x_i, A_{i+i'} = x_{i+i'})
\end{aligned} \tag{4.7}$$

Similarly, $\nabla C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) = Pr(A^l_i = x_i, A^l_{i+i'} = x_{i+i'})$. □

Theorem 13. *The HoMMPP copula has recursive relationship between* $C^l_{i'}$ *and* $C^{l-1}_{i'}$ *as shown below:*

$$C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) = \sum_{x=0}^{x_i} \sum_{y=0}^{x_{i+i'}} C^{l-1}_{i'}(M^{l-1}(x_i - x), M^{l-1}(x_{i+i'} - y)) * \nabla C_{i'}(M(x), M(y)), \tag{4.8}$$

Proof. Since the constituent MMPPs are mutually independent, the arrivals of l number of aggregate MMPPs can be divided into arrivals of $(l-1)$ number of aggregate MMPPs plus a single MMPP arrivals, *i.e.*, $A^l_i = A^{l-1}_i + A_i$. Following this idea, we

have:

$$
\begin{aligned}
& C^l_{i'}(M^l(x_i), M^l(x_{i+i'})) \\
=& Pr(A^l_i \le x_i, A^l_{i+i'} \le x_{i+i'}) \\
=& \sum_{x=0}^{x_i} \sum_{y=0}^{x_{i+i'}} Pr(A^l_i \le x_i, A^l_{i+i'} \le x_{i+i'} | A_i = x, A_{i+i'} = y) * Pr(A_i = x, A_{i+i'} = y) \\
=& \sum_{x=0}^{x_i} \sum_{y=0}^{x_{i+i'}} Pr(A^{l-1}_i \le x_i - x, A^{l-1}_{i+i'} \le x_{i+i'} - y) * Pr(A_i = x, A_{i+i'} = y) \\
=& \sum_{x=0}^{x_i} \sum_{y=0}^{x_{i+i'}} C^{l-1}_{i'}(M^{l-1}(x_i - x), M^{l-1}(x_{i+i'} - y)) * \nabla C_{i'}(M(x), M(y)).
\end{aligned}
$$

□

Even with Theorem 13, the closed form of HoMMPP copula $C^l_{i'}$ can hardly be derived. However the recursive relationship between $C^l_{i'}$ and $C^{l-1}_{i'}$ can be implemented as a recursive algorithm to calculate HoMMPP copula values numerically. The algorithm will be introduced in Section 4.4.4

4.4.3 Theoretical Copula Analysis for HeMMPP

HeMMPP is very similar to HoMMPP in their definitions, except that the constituent MMPPs in HeMMPP are different rather than the same. Thus we have to differentiate the constituent MMPPs by numbering them. With a shuffling, we can get a random order of constituent MMPPs, *i.e.*, $({}_1Q, {}_1\Lambda), ({}_2Q, {}_2\Lambda), ..., ({}_lQ, {}_l\Lambda)$, where $({}_lQ, {}_l\Lambda)$ represents the parameters of the l-th constituent MMPP. **The notations for each constituent MMPP will be labeled by the order value l on the bottom left**, for instance, ${}_lA_i$, ${}_lC_{i'}$, ${}_lM$ are arrival counts, copula, marginal of l-th MMPP. In HeMMPP, A^l_i, $C^l_{i'}$, M^l denote those notations of the superposition of the first l number of constituent MMPPs. Note that we introduce this ordering for a clear explaining and analysis.

We derive the following theorems to analyse the marginal distribution and the copula of HeMMPP:

Theorem 14. *The HeMMPP marginal distribution function has recursive relation-*

ship between M^l and M^{l-1} as

$$M^l(x_i) = \sum_{x=0}^{x_i} M^{l-1}(x_i - x) * {}_lp(x), \tag{4.9}$$

where ${}_lp$ is the probability mass function of the arrival count from l-th MMPP, ${}_lp(x) = {}_lM(x) - {}_lM(x-1)$.

Proof. The key idea of the proof is to divide the arrival from l number of MMPPs into the arrival from the first $l-1$ number of MMPPs plus that from the l-th MMPP, i.e., $A_i^l = A_i^{l-1} + {}_lA_i$. Thus, we have

$$\begin{aligned} M^l(x_i) &= Pr(A_i^l \le x_i) = \sum_{x=0}^{x_i} Pr(A_i^l \le x_i | {}_lA_i = x) Pr({}_lA_i = x) \\ &= \sum_{x=0}^{x_i} Pr(A_i^{l-1} \le x_i - x) Pr({}_lA_i = x) = \sum_{x=0}^{x_i} M^{l-1}(x_i - x) * {}_lp(x) \end{aligned}$$

□

Theorem 15. *The HeMMPP copula has the recursive relationship between $C_{i'}^l$ and $C_{i'}^{l-1}$ as shown below:*

$$C_{i'}^l(M^l(x_i), M^l(x_{i+i'})) = \sum_{x=0}^{x_i} \sum_{y=0}^{x_{i+i'}} C_{i'}^{l-1}(M^{l-1}(x_i - x), M^{l-1}(x_{i+i'} - y)) * \nabla_l C_{i'}(M(x), M(y)), \tag{4.10}$$

where $\nabla_l C_{i'}$ is the single MMPP copula gradient of the l-th MMPP.

Proof. The proof is omitted since it is just similar to that of Theorem 13 on the basis of $A_i^l = A_i^{l-1} + {}_lA_i$. □

4.4.4 An Algorithm to Compute HeMMPP Copula

In Sections 4.4.2 and 4.4.3, we introduce the recursive relationships among HoMMPP/ HeMMPP copula. Although the HoMMPP/ HeMMPP copulas are not derived into closed forms, they could be computed numerically with a recursive algorithm introduced in this section. Since both single MMPP and HoMMPP are special cases of HeMMPP, we will introduce how our algorithm works on HeMMPP as a general case.

Consider HeMMPP with l number of heterogeneous constituent MMPPs. To limit the running time of the algorithm, we narrow down the interested range of A_i^l from its infinite domain to finite range with an upper threshold $\hat{a}$. In other words, although the range of A_i^l is on the whole non-negative integer domain, we are only interested in computing marginal values $M^l(x_i)$ and copula values $C_{i'}^l(M^l(x_i), M^l(x_{i+i'}))$ for $x_i < \hat{a}$ and $x_{i+i'} < \hat{a}$. The selection of $\hat{a}$ is application dependent and can be set appropriately based on observations. Narrowing down the interested range makes the computation feasible and still fulfills the demand for real applications, because the observations of arrival counts in real traffic flows always fall within a limited range.

On the interested range $[0, \hat{a})$, we define three matrices in Table 4.1, $\mathbb{M}^l$ to represent HeMMPP/HoMMPP marginal values, $\mathbb{C}^l$ to represent HeMMPP/HoMMPP copula values, and $\mathbb{D}^l$ to represent HeMMPP/HoMMPP copula gradient values. Essentially, these three matrices are look-up tables for HeMMPP on the domain of interested range $[0, \hat{a})$. For constituent MMPPs, their values in PMF, CDF marginal, copula and copula gradient are represented by matrices ${}_l\mathbb{P}$, ${}_l\mathbb{M}$, ${}_l\mathbb{C}$, ${}_l\mathbb{D}$, where l means the order of constituent MMPP.

Note that $\mathbb{C}^l$ is defined for copula between A_i^l and $A_{i+i'}^l$, as time lag i' is set to a certain constant. Similarly, $\mathbb{C}^l$, $\mathbb{D}^l$, ${}_l\mathbb{C}$ and ${}_l\mathbb{D}$ are defined under condition that i' is preset as a constant. To emphasize the matrices' dimension, we mark dimensions on the bottom right, such as $[\mathbb{M}^l]_{\hat{a}}$,$[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$ etc. We also define notations for submatrix, for instance, $[\mathbb{C}^l]_{x\times y}$ to represent the submatrix of $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$ with its first x rows and first y columns.

With HeMMPP parameters $({}_1Q, {}_1\Lambda), ({}_2Q, {}_2\Lambda), ..., ({}_lQ, {}_l\Lambda)$ and a properly set threshold value $\hat{a}$, we design Algorithm 2 (with the time complexity as $\mathcal{O}(\hat{a} \times l)$) to calculate HeMMPP marginal matrix $[\mathbb{M}^l]_{\hat{a}}$ and Algorithm 3 (with the time complexity as $\mathcal{O}(\hat{a} \times \hat{a} \times l)$) to calculate HeMMPP copula matrix $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$. In Algorithm 2, the recursive relationship in Theorem 14 is implemented as the procedure MarginalMatrixCalc. Some details in this procedure are expanded here:

- In line 6 and line 11, the marginal matrix is calculated for single MMPP. Given l-th constituent MMPP $({}_lQ, {}_l\Lambda)$, its stationary distribution ${}_l\Pi$ and conditional marginal vector ${}_l\mathbb{G}$ could be calculated from parameters according to Theorem 10. Then the element of its marginal matrix is calculated as ${}_l\mathbb{M}_x = {}_l\Pi_l\mathbb{G}(x-1)$ as shown in Theorem 10;

- In line 12, $[{}_l\mathbb{P}]_{\hat{a}}$ is computed from $[{}_l\mathbb{M}]_{\hat{a}}$ by ${}_l\mathbb{P}_x = {}_l\mathbb{M}_x - {}_l\mathbb{M}_{x-1}$ for any x.

Table 4.1: Definition of Matrices

Matrix Denotation	Matrix name	Number in row x (and column y)
$[\mathbb{M}^l]_{\hat{a}}$	HeMMPP marginal matrix	$\mathbb{M}^l_x \equiv M^l(x-1) = Pr(A^l_i \leq x-1)$
$[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$	HeMMPP copula matrix	$\mathbb{C}^l_{xy} \equiv C^l_{i'}(M^l(x-1), M^l(y-1))$ $= Pr(A^l_i \leq x-1, A^l_{i+i'} \leq y-1)$
$[\mathbb{D}^l]_{\hat{a}\times\hat{a}}$	HeMMPP copula gradient matrix	$\mathbb{D}^l_{xy} \equiv \nabla C^l_{i'}(M^l(x-1), M^l(y-1))$ $= Pr(A^l_i = x-1, A^l_{i+i'} = y-1)$
$[{}_l\mathbb{M}]_{\hat{a}}$	l-th MMPP marginal matrix	${}_l\mathbb{M}_x \equiv {}_lM(x-1) = Pr({}_lA_i \leq x-1)$
$[{}_l\mathbb{P}]_{\hat{a}}$	l-th MMPP PMF matrix	${}_l\mathbb{P}_x \equiv {}_lp(x-1) = Pr({}_lA_i = x-1)$
$[{}_l\mathbb{C}]_{\hat{a}\times\hat{a}}$	l-th MMPP copula matrix	${}_l\mathbb{C}_{xy} \equiv {}_lC_{i'}({}_lM(x-1), {}_lM(y-1))$ $= Pr({}_lA_i \leq x-1, {}_lA_{i+i'} \leq y-1)$
$[{}_l\mathbb{D}]_{\hat{a}\times\hat{a}}$	l-th MMPP copula gradient matrix	${}_l\mathbb{D}_{xy} \equiv \nabla_l C_{i'}({}_lM(x-1), {}_lM(y-1))$ $= Pr({}_lA_i = x-1, {}_lA_{i+i'} = y-1)$

Similarly, Algorithm 3 implements Theorem 15 via a recursive procedure called CopulaMatrixCalc:

- In line 6 and line 11, the marginal matrix is calculated for single MMPP. Given l-th constituent MMPP $({}_lQ, {}_l\Lambda)$, its stationary distribution ${}_l\Pi$, conditional marginal vector ${}_l\mathbb{G}$ and transition matrix ${}_lP(i'\Delta)$ could be calculated from parameters. Then the element of its copula matrix is calculated as ${}_l\mathbb{C}_{xy} = {}_l\mathbb{G}(x-1)diag({}_l\Pi){}_lP(i'\Delta){}_l\mathbb{G}(y-1)^T$ according to Theorem 11;
- In line 12, $[{}_l\mathbb{D}]_{\hat{a}\times\hat{a}}$ is computed from $[{}_l\mathbb{C}]_{\hat{a}\times\hat{a}}$ by ${}_l\mathbb{D}_{xy} = {}_l\mathbb{C}_{xy} + {}_l\mathbb{C}_{(x-1)(y-1)} - {}_l\mathbb{C}_{(x-1)y} - {}_l\mathbb{C}_{x(y-1)}$ for any x and y.

With Algorithm 2 and 3, marginal matrix $[\mathbb{M}^l]_{\hat{a}}$ and copula matrix $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$ are calculated as the numerical results of HeMMPP marginal distributions and its temporal copula as summarized in the following theorem:

Theorem 16. ***(HeMMPP copula)*** *Given HeMMPP with marginal matrix* $[\mathbb{M}^l]_{\hat{a}}$ *and copula matrix* $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$*, its copula value of* $C^l_{i'}(u_i, u_{i+i'})$ *for any* $u_i \leq \mathbb{M}^l_{\hat{a}}$ *and* $u_{i+i'} \leq \mathbb{M}^l_{\hat{a}}$ *will be calculated as steps:*

Algorithm 2 An algorithm to compute HeMMPP marginal matrix $\mathbb{M}^l$

Require: HeMMPP parameters $({}_1Q,{}_1\Lambda), ({}_2Q,{}_2\Lambda), ..., ({}_lQ,{}_l\Lambda)$, the upper threshold $\hat{a}$
Ensure: $[\mathbb{M}^l]_{\hat{a}}$

```
return MargalMatrixCalc([1Λ, ..., lΛ], [1Q, ..., lQ], â)

procedure MarginalMatrixCalc([1Λ, ..., lΛ], [1Q, ..., lQ], â)
    l ← the vector length of [1Λ, ..., lΛ] or of [1Q, ..., lQ]
    // Base Case
    if l == 1 then
        [M^1]_â ← compute with parameters 1Λ and 1Q based on Theorem 10
        return [M^1]_â
    end if
    // Inductive Step
    [M^{l-1}]_â ← MarginalMatrixCalc([1Λ, ..., l−1Λ], [1Q, ..., l−1Q], â)
    [lM]_â ← compute with parameters lΛ and lQ based on Theorem 10
    [lP]_â ← compute from [lM]_â
    for x ← 1, â do
        Rotate matrix [lP]_x 180 degree clockwise as [lP']_x
        Calculate Hadamard product of [M^{l-1}]_x and [lP']_x as [T]_x
        M^l_x ← sum of all elements in matrix [T]_x
    end for
    return [M^l]_â
end procedure
```

1. $x_i = (\text{argmax}_x\, \mathbb{M}^l_x \leq u_i) - 1$;
2. $x_{i+i'} = (\text{argmax}_x\, \mathbb{M}^l_x \leq u_{i+i'}) - 1$;
3. $C^l_{i'}(u_i, u_{i+i'}) = \mathbb{C}^l_{(x_i+1)(x_{i+i'}+1)}$.

For short, $C^l_{i'}(u_i, u_{i+i'}) = \mathbb{C}^l_{(\text{argmax}_x\, \mathbb{M}^l_x \leq u_i)(\text{argmax}_x\, \mathbb{M}^l_x \leq u_{i+i'})}$

With all the analysis in this section, we find the way to calculate the copula for HeMMPP as shown in Theorem 16. Although mathematically it is not in closed-form, the copula values can be computed effectively. Therefore, Algorithm 2 and 3 can be regarded as the theoretical analysis of HeMMPP copula, and offer the exact solution for the temporal dependence.

Algorithm 3 An algorithm to compute HeMMPP copula matrix $\mathbb{C}^l$

Require: HeMMPP parameters $({}_1Q, {}_1\Lambda), ({}_2Q, {}_2\Lambda), ..., ({}_lQ, {}_l\Lambda)$, the upper threshold $\hat{a}$
Ensure: $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$

return CopulaMatrixCalc$([{}_1\Lambda, ..., {}_l\Lambda], [{}_1Q, ..., {}_lQ], \hat{a})$

procedure CopulaMatrixCalc$([{}_1\Lambda, ..., {}_l\Lambda], [{}_1Q, ..., {}_lQ], \hat{a})$
  $l \leftarrow$ the vector length of $[{}_1\Lambda, ..., {}_l\Lambda]$ or of $[{}_1Q, ..., {}_lQ]$
  // Base Case
  if $l == 1$ **then**
    $[\mathbb{C}^1]_{\hat{a}\times\hat{a}} \leftarrow$ compute with parameters ${}_1\Lambda$ and ${}_1Q$ based on Theorem 11
    return $[\mathbb{C}^1]_{\hat{a}\times\hat{a}}$
  end if
  // Inductive Step
  $[\mathbb{C}^{l-1}]_{\hat{a}\times\hat{a}} \leftarrow$ CopulaMatrixCalc$([{}_1\Lambda, ..., {}_{l-1}\Lambda], [{}_1Q, ..., {}_{l-1}Q], \hat{a})$
  $[{}_l\mathbb{C}]_{\hat{a}\times\hat{a}} \leftarrow$ compute with parameters ${}_l\Lambda$ and ${}_lQ$ based on Theorem 11
  $[{}_l\mathbb{D}]_{\hat{a}\times\hat{a}} \leftarrow$ compute from $[{}_l\mathbb{C}]_{\hat{a}\times\hat{a}}$
  for $x \leftarrow 1, \hat{a}$ **do**
    for $y \leftarrow 1, \hat{a}$ **do**
      Rotate matrix $[{}_l\mathbb{D}]_{x\times y}$ 180 degree clockwise to be $[{}_l\mathbb{D}']_{x\times y}$
      Calculate Hadamard product of $[\mathbb{C}^{l-1}]_{x\times y}$ and $[{}_l\mathbb{D}']_{x\times y}$ as $[\mathbb{T}]_{x\times y}$
      $\mathbb{C}^l_{xy} \leftarrow$ sum of all elements in matrix $[\mathbb{T}]_{x\times y}$
    end for
  end for
  return $[\mathbb{C}^l]_{\hat{a}\times\hat{a}}$
end procedure

4.5 Parametric Copula Modeling for MMPP trace

Parametric copula modeling is to fit trace to well known parametric copulas and choose the best one for applications. The arrival count traces could be from any kind of MMPPs: single MMPP, HoMMPP or HeMMPP. The marginal distribution will be constructed empirically and parametric copulas can be chosen according to the tail dependence. In general, we assume that the fitting trace, denoted as $\{x_i\}_{1\le i\le n}$, is a sample trace from HeMMPP $\{A^l_i\}$. Our goal is to model copula between A^l_i and $A^l_{i+i'}$. We proposed the following tail-dependence-based schema to conduct parametric copula modeling:

1. Compute the tail dependence from data.
   The upper tail dependence, as the limit of a function as u approaches 1, can be approximated by evaluating a function value at u where u is close to 1 [83], say

0.99. Similarly, the lower tail dependence can be approximated by evaluating the function value at u where u is close to 0, say 0.01, *i.e.*,

$$\begin{aligned}\rho_t^+ &\approx Pr(X > F_X^{-1}(u)|Y > F_Y^{-1}(u))|_{u=0.99} \approx \frac{1-2u+C(u,u)}{1-u}|_{u=0.99} \\ \rho_t^- &\approx Pr(X < F_X^{-1}(u)|Y < F_Y^{-1}(u))|_{u=0.01} \approx \frac{C(u,u)}{u}|_{u=0.01}.\end{aligned} \tag{4.11}$$

The tail dependence between A_i^l and $A_{i+i'}^l$ is estimated from trace as follows

$$\begin{aligned}\rho_t^+ &\approx Pr(A_i^l > x^+|A_{i+i'}^l > x^+) = \frac{\sum_{i=1}^{n-i'} \mathbf{1}(x_i > x^+, x_{i+i'} > x^+)}{\sum_{i=1}^{n-i'} \mathbf{1}(x_{i+i'} > x^+)} \\ \rho_t^- &\approx Pr(A_i^l < x^-|A_{i+i'}^l < x^-) = \frac{\sum_{i=1}^{n-i'} \mathbf{1}(x_i < x^-, x_{i+i'} < x^-)}{\sum_{i=1}^{n-i'} \mathbf{1}(x_{i+i'} < x^-)}.\end{aligned} \tag{4.12}$$

where x^+ and x^- are high and low quantile values such that $\hat{M}^l(x^+) = 0.99$ and $\hat{M}^l(x^-) = 0.01$, and $\hat{M}^l$ is the empirical marginal distribution of A_i^l.

2. Choose one candidate copula based on tail dependence property.
   Choose proper copula in the candidate set according to tail dependence, for instance, we could choose Clayton copula if $\rho_t^+ \approx 0$ and $\rho_t^- > 0$; choose Gumbel copula if $\rho_t^+ > 0$ and $\rho_t^- \approx 0$; choose Frank copula if $\rho_t^+ \approx \rho_t^-$; or use any mixtures of these three copulas, the mixtures will cover various tail dependences.

3. Fit data to determine the copula parameter.
   Each observation of the sample trace is first evaluated in its marginal domain, that is, $u_i = \hat{M}^l(x_i)$. Then the pairs of $\{(u_i, u_{i+i'})\}_{1 \le i \le n-i'}$ become the data to fit to determine the copula parameter θ. The fitting is implemented by maximum likelihood estimation method explained in details in [12]. The parametric copula learned from MMPP trace is denoted as $C(u_i, u_{i+i'}; \theta)$.

4.6 Summary

This chapter theoretically derive the intricate temporal dependence structure in MMPPs with copula analysis. It presents the theoretical solution for modeling temporal dependence in both single MMPP and HoMMPP/ HeMMPP. In addition, parametric copula modeling schema has been proposed for MMPP traces. In the next three chapters, we will apply the analytical results in this chapter under different scenarios.

Chapter 5

Application of MMPP Copulas for Network Traffic Prediction

In this chapter, we apply MMPP copulas discussed in Chapter 4 for prediction of network traffic flows.

5.1 Introduction

Nowadays, people rely heavily on the Internet and various digital platforms supported by enterprise cloud-computing capabilities, where data volume from online banking, video broadcast, and social networking increases at an unprecedented pace. The huge amount and diverse patterns of Internet traffic require large enterprises and service providers to develop a new spectrum of technologies for serving their customers easily, quickly and with guaranteed quality of service (QoS). To face the challenge, some large enterprises have started to explore the power of predictive resource provisioning so that resource allocation aligns well with the dynamic service demands [8]. A good prediction on traffic flow will benefit the service provisioning.

The network traffic flows can be regarded as time series. The prediction of network traffic flows can be made based on some existing methods, for instance, linear predictive coding and autoregressive model. Different from these existing methods, copula modeling characterize the full temporal dependence among network traffic, and will benefit the prediction in several aspects:

- Copula can capture various temporal dependence. With either the theoretical copula or plenty of parametric copulas to choose, a variety of temporal depen-

dencies can be modeled. In other words, the copula modeling provides us with numerous choices of temporal dependence to model real-world network traffic;

- The invariant property of copula makes copula model stable when functional changes occur on network traffic. Without a re-modeling process, copula-based prediction will be as precise as before changing, while the other existing models can't guarantee it.

In this chapter, we conduct prediction on MMPP traffic flows. Both theoretical copulas derived and parametric copulas modeling proposed in Chapter 4 are used to build the temporal dependence and predict future trend of traffic flows. With a large number of prediction on real-world traces and simulations, we show that copula-based prediction outperforms classical prediction models, linear predictive coding model and autoregressive model.

5.2 Copula-based Prediction

The problem of traffic prediction can be posed in different forms. In our work, we focus on estimating the future arrival count $A_{i+i'}$ based on the current observation of arrival count A_i. The prediction is made by maximizing the conditional probability $Pr(A_{i+i'}|A_i)$, *i.e.*, $\hat{x}_{i+i'} = \text{argmax}_x \, Pr(A_{i+i'} = x|A_i = x_i)$. When $i' = 1$, the prediction is made one-step forward; when $i' > 1$, the prediction is made multi-step forward. In this section, we introduce the prediction method with theoretical copulas, followed by a discussion for prediction with parametric copulas.

5.2.1 Prediction Based on Theoretical Copulas

With MMPP copula $C_{i'}$ for single MMPP and theoretical copula $C^l_{i'}$ for HoMMPP/ HeMMPP, Theorem 17 can be used to predict future arrivals.

Theorem 17. *(1) Consider a MMPP having its copula $C_{i'}$ between A_i and $A_{i+i'}$. If $A_i = x_i$ is the current observation from the arrival process and if the prediction is made by maximizing the conditional probability $Pr(A_{i+i'}|A_i)$, the predicted arrival count $\hat{x}_{i+i'}$ is:*

$$\hat{x}_{i+i'} = \underset{x}{\text{argmax}} \, \nabla C_{i'}(M(x_i), M(x)). \tag{5.1}$$

(2) Consider a HoMMPP/HeMMPP having theoretical copula $C^l_{i'}$ between A^l_i and $A^l_{i+i'}$. If $A^l_i = x$ is the current observation from the arrival process and if the prediction

is made by maximizing the conditional probability $Pr(A^l_{i+i'}|A^l_i)$*, the predicted arrival count* $\hat{x}_{i+i'}$ *is:*

$$\hat{x}_{i+i'} = \operatorname*{argmax}_x \nabla C^l_{i'}(M^l(x_i), M^l(x)). \tag{5.2}$$

Proof. We only prove part (2), since part (1) is a special case of part (2). Since the prediction is made by maximizing the conditional probability $Pr(A^l_{i+i'}|A^l_i)$, we have

$$\begin{aligned}
\hat{x}_{i+i'} &= \operatorname*{argmax}_x Pr(A^l_{i+i'} = x | A^l_i = x_i) \\
&= \operatorname*{argmax}_x \frac{Pr(A^l_i = x_i, A^l_{i+i'} = x)}{Pr(A^l_i = x_i)} \\
&= \operatorname*{argmax}_x Pr(A^l_i = x_i, A^l_{i+i'} = x) \\
&= \operatorname*{argmax}_x \nabla C^l_{i'}(M^l(x_i), M^l(x))
\end{aligned}$$

□

According to the definitions in Table 4.1, the value of $\nabla C^l_{i'}$ function is represented by HeMMPP/HoMMPP copula gradient matrix $\mathbb{D}^l$. With this numerical transformation between the function and matrix, the predicted arrival count is

$$\hat{x}_{i+i'} = \operatorname*{argmax}_x \mathbb{D}^l_{(x_i+1)(x+1)} = (\operatorname*{argmax}_x \mathbb{D}^l_{(x_i+1)x}) - 1.$$

It indicates that the predicted arrival count can be numerically determined as the column number of maximum value in the $(x_i + 1)$-th row of the matrix $\mathbb{D}^l$ minus 1.

5.2.2 Prediction Based on Parametric Copulas

Given a single MMPP or HoMMPP/HeMMPP trace $\{x_i\}$, parametric copula modeling is conducted according to Section 4.5. The parametric copula is continuous on the domain of $[0, 1]$, however, the marginal distribution is discrete. Due to this reason, we first study the prediction problem on stochastic processes with continuous marginals (Theorem 18) and then extend its usage for discrete distributions (Theorem 19).

Theorem 18. *Consider a stochastic process* $\{B_i\}$ *that has a parametric copula* $C(u_i, u_{i+i'}; \theta)$ *between* B_i *and* $B_{i+i'}$*, continuous marginal distribution* F*, and marginal probability density function (PDF)* f*. We have the following the conditional PDF as*

$$f(B_{i+i'} = x_{i+i'}|B_i = x_i) = c(F(x_i), F(x_{i+i'}); \theta) f(x_{i+i'}), \tag{5.3}$$

where $c(u, v; \theta) = \frac{\partial}{\partial u}\frac{\partial}{\partial v}C(u, v; \theta)$ *is called the parametric copula density function.*

For discrete marginals, we revise Theorem 18 by relating the probability density function(PDF) in continuous distribution to the probability mass function(PMF) in discrete distribution.

Theorem 19. *Consider a statistic process* $\{B_i\}$ *that has a parametric copula* $C(u_i, u_{i+i'}; \theta)$ *between* B_i *and* $B_{i+i'}$*, discrete marginal distribution* F*, and marginal probability mass function (PMF)* p*. We have the following the conditional pmf as*

$$p(B_{i+i'} = x_{i+i'}|B_i = x_i) = c(F(x_i), F(x_{i+i'}); \theta)p(x_{i+i'}). \tag{5.4}$$

The proofs of Theorems 18 and 19 are straightforward using similar techniques in [5]. With Theorem 19, prediction based on parametric copula on a MMPP trace is given by Theorem 20.

Theorem 20. *(1) Consider a MMPP having its parametric copula* $C(u_i, u_{i+i'}; \theta)$ *between* A_i *and* $A_{i+i'}$*. If* $A_i = x_i$ *is the current observation from the arrival process and if the prediction is made by maximizing the conditional probability* $Pr(A_{i+i'}|A_i)$*, the predicted arrival count* $\hat{x}_{i+i'}$ *is:*

$$\hat{x}_{i+i'} = \operatorname*{argmax}_x c(M(x_i), M(x); \theta)(M(x) - M(x-1)); \tag{5.5}$$

(2) Consider a HoMMPP/HeMMPP having its parametric copula $C(u_i, u_{i+i'}; \theta)$ *between* A_i^l *and* $A_{i+i'}^l$*. If* $A_i^l = x_i$ *is the current observation from the arrival process and if the prediction is made by maximizing the conditional probability* $Pr(A_{i+i'}^l|A_i^l)$*, the predicted arrival count* $\hat{x}_{i+i'}$ *is:*

$$\hat{x}_{i+i'} = \operatorname*{argmax}_x c(M^l(x_i), M^l(x); \theta)(M^l(x) - M^l(x-1)). \tag{5.6}$$

5.3 Experimental Evaluation

We conduct experiments to show how the copula model could help traffic prediction. In the evaluation, we first give a broad view of the methods to evaluate the per-

formance of copula-based prediction. We then show case studies on single MMPP, HoMMPP and HeMMPP.

5.3.1 Evaluation Methods

Theoretical copula and parametric copulas discussed in Chapter 4 will be used for traffic prediction according to Section 5.2. To evaluate copula models for prediction, we implement two classic prediction models, autoregressive model (AR(1)) and linear predictive coding (LPC(1)), for comparison. Note that the first order AR model and the first order LPC model are used here for a fair comparison, because our copula-based prediction model is first order in the sense that only dependence between two successive arrival counts is considered each time.

1. AR(1) model prediction
   Consider a trace having AR(1) model with parameters φ_1, φ_2 and white noise ϵ_t. If $A_i^l = x_i$ is the current observation, the prediction is made by:

$$\hat{x}_{i+1} = \varphi_1 + \varphi_2 x_i + \epsilon_{i+1},$$
$$\hat{x}_{i+2} = \varphi_1 + \varphi_2 \hat{x}_{i+1} + \epsilon_{i+2},$$
$$\cdots$$
$$\hat{x}_{i+i'} = \varphi_1 + \varphi_2 \hat{x}_{i+i'-1} + \epsilon_{i+i'},$$

2. LPC(1) model prediction
   Consider a trace having LPC(1) model with the parameter σ. If $A_i^l = x_i$ is the current observation, the prediction is made by:

$$\hat{x}_{i+1} = \sigma x_i,$$
$$\hat{x}_{i+2} = \sigma \hat{x}_{i+1},$$
$$\cdots$$
$$\hat{x}_{i+i'} = \sigma \hat{x}_{i+i'-1},$$

As a purely linear predictor, the parameter of LPC(1), σ is directly determined by auto-correlation of arrival count sequence. Since LPC(1) model is to predict data only based on the dependence information in terms of autocorrelation, it is set as the benchmark predictor to show how functional dependence modeling with copulas

improves over linear dependence. We also compare copula-based prediction with AR(1) model since AR(1) model is the popular statistical method for prediction.

When applying any of the prediction models on a traffic trace, the trace is divided into two parts, the training set and the testing set. The training set comes from the first certain percentage of trace data, and the rest of the trace constitutes the testing set. For example, if the training percentage is 50%, the first half of the trace will be used to train a model, and the second half will be used to test prediction accuracy. The prediction accuracy is measured by root-mean-square error (RMSE) across the test set, defined as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}, \tag{5.7}$$

where x_i is the i-th observed arrival count from test set, $\hat{x}_i$ denotes the corresponding predicted value, and n is the total number of time slots in the testing period.

For a prediction model, its average RMSE (aRMSE) over different experiment scenarios represents its overall performance on MMPP traffic trace prediction. Its performance improvement ratio (IMP RATIO) over benchmark model (LPC(1)) is defined in Eq.(5.8). The larger the value is, the more the predictor improves over LPC(1) model.

$$\text{IMP RATIO} = \frac{\text{aRMSE}_{\text{benchmark}} - \text{aRMSE}}{\text{aRMSE}_{\text{benchmark}}} * 100\%. \tag{5.8}$$

5.3.2 Case Study on A Single MMPP Trace from Real-world

BCpAug89 trace, one of Bellcore traces[1], records the exact arrival times of 1,000,000 packets on an Ethernet at Bellcore Morristown Research and Engineering facility. Previous research has shown that the trace is well characterized by MMPP [3, 62, 58]. We analyse the trace in terms of arrival counts every second, *i.e.*, the length of time slot is set as $\Delta = 1$ (second), and A_i denotes random variable of arrival count in i-th second. With learning algorithm proposed in [39], this trace is modeled by a 12-state MMPP with parameters $({}_AQ, {}_A\Lambda)$ as shown in Eq. (5.9). In the case study, we will apply copula to model its dependence structure and predict the trace flow. We will also vary the trace by a functional transformation to show that the copula-based dependence model is much more stable than other models.

[1]The Bellcore traces are available on the website http://ita.ee.lbl.gov/html/contrib/BC.html

$$
{}_AQ = \begin{pmatrix}
-0.857 & 0.286 & 0.428 & 0.143 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.067 & -0.900 & 0.267 & 0.233 & 0.233 & 0.067 & 0.033 & 0 & 0 & 0 & 0 & 0 \\
0.023 & 0.078 & -0.837 & 0.336 & 0.203 & 0.103 & 0.078 & 0 & 0.016 & 0 & 0 & 0 \\
0 & 0.026 & 0.140 & -0.722 & 0.274 & 0.153 & 0.085 & 0.030 & 0.007 & 0.007 & 0 & 0 \\
0.002 & 0.008 & 0.051 & 0.173 & -0.651 & 0.244 & 0.122 & 0.041 & 0.006 & 0.002 & 0.002 & 0 \\
0 & 0.001 & 0.027 & 0.074 & 0.173 & -0.696 & 0.303 & 0.094 & 0.014 & 0.009 & 0.001 & 0 \\
0 & 0.001 & 0.004 & 0.019 & 0.099 & 0.233 & -0.617 & 0.200 & 0.048 & 0.012 & 0.001 & 0 \\
0 & 0 & 0.008 & 0.023 & 0.049 & 0.184 & 0.409 & -0.775 & 0.084 & 0.015 & 0.003 & 0 \\
0 & 0 & 0.008 & 0.015 & 0.015 & 0.120 & 0.301 & 0.218 & -0.805 & 0.113 & 0.015 & 0 \\
0 & 0.020 & 0 & 0 & 0.059 & 0.059 & 0.235 & 0.078 & 0.275 & -0.824 & 0.098 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.077 & 0.231 & 0.231 & 0.154 & 0.077 & -0.847 & 0.077 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1
\end{pmatrix},
$$

$$
{}_A\Lambda = (782.069, 674.207, 574.345, 482.483, 398.621, 322.759, 254.897, 195.035, 143.173, 99.311, 63.449, 35.587). \tag{5.9}
$$

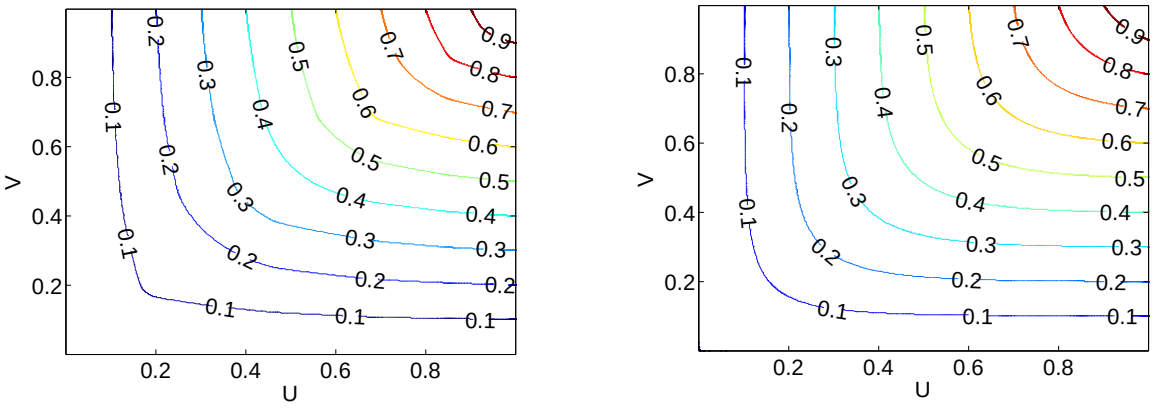
One-step Prediction on BCpAug89 trace

Given the learned MMPP parameter $({}_AQ, {}_A\Lambda)$, we first construct copula for MMPP theoretically and empirically. With theoretical analysis in Section 4.4.1, the MMPP-copula for learned MMPP is computed from the parameters $({}_AQ, {}_A\Lambda)$ based on Theorem 11. The contour of the computed MMPP copula is shown in Fig 5.1a. Dependence measures, including Kendall's tau ρ_τ, Spearman's rho ρ_s, tail dependence ρ_t^+ and ρ_t^-, and Pearson correlation coefficient ρ, between A_i and A_{i+1} are analysed in Table 5.1. Theoretical results of ρ_τ, ρ_s and ρ_t are calculated from copula with Eqs.(2.2)-(2.5). Pearson coefficient is calculated based on the analysis in [61]. Except Pearson coefficient, all other dependence measures can be obtained via copula, indicating that copula includes rich information about dependence structure. In addition, the comparison between theoretical and empirical dependence measures shows that copula accurately captures the trace dependence.

Table 5.1: Dependence Measures of BCpAug89 Trace from Theoretical Analysis and Empirical Analysis

	ρ_τ	ρ_s	$\rho_t^+\|_{u=0.99}$	$\rho_t^-\|_{u=0.01}$	ρ
Theoretical	0.4788	0.6150	0.4067	0.3359	0.7555
Empirical	0.4212	0.5897	0.3935	0.3248	0.6149

Empirically, a parametric copula can be also chosen to model temporal dependence



(a) Theoretical MMPP copula.

(b) Parametric copula.

Figure 5.1: Copula contours for MMPP learned from BCpAug89 trace.

between A_i and A_{i+1}. As shown in Table 5.1, ρ_t^+ is close to ρ_t^-, we thus choose Frank copula to model the BCpAug89 trace. The parameter of Frank copula is determined by fitting a training set from the trace. With different training percentage of data, the parameter of Frank copula will be determined accordingly. Fig. 5.1b shows the contour of parametric copula trained from 80% percentage of data in BCpAug89 trace. Fig. 5.1a and Fig. 5.1b have close contour shape. The similarity of two copulas can be quantified by the discrete L_2 norm distance over the size of discrete lattice [27], which is 0.0173 in our case. This value is close to those in the experiments of [27] when selecting two similar copulas, indicating that the parametric copula trained from data accords well with the theoretical copula from analysis.

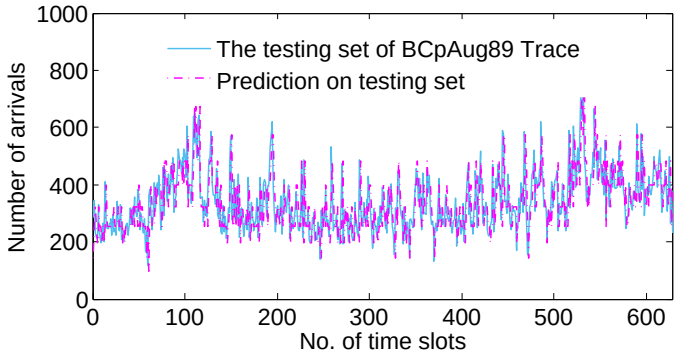


Figure 5.2: Prediction with theoretical copula on the testing set (last 20%) of BCpAug89 trace

With the copulas constructed from the training set of BCpAug89 trace, one-step prediction is conducted on its testing set. Fig. 5.2 shows at a glance the prediction

Table 5.2: One-Step Prediction RMSE on BC-pAug89 trace with Different Training Percentages.

Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	94.2411	88.9070	92.2850	110.3130
60%	90.5974	87.5581	88.8550	106.1805
70%	93.1982	91.7414	90.8745	108.5895
80%	92.3244	88.1251	90.2618	105.2930
90%	94.4256	93.2204	92.9318	108.0254
aRMSE	92.95734	89.9104	91.04162	107.6803
IMP RATIO	13.67%	16.50%	15.45%	—

with theoretical copula on the last 20% arrivals of BCpAug89 trace. To obtain multiple prediction results for the aRMSE measurement, we adjust the training percentage from 50% to 90%. The prediction accuracy of copulas in measure of RMSE is shown in detail and compared with AR(1) model and LPC(1) model in Table 5.2. From the table, we can infer that both MMPP copula and parametric copula characterize the temporal dependence of BCpAug89 trace well, leading to a good prediction. Copula-based predictions, including theoretical copula model and parametric copula model, have more than 10% improvement ratio over the LPC(1) model, showing the advantage of functional dependence modeling (such as copulas) over linear dependence measurement (such as autocorrelation). Copula-based predictions achieve accuracy similar to the classical AR(1) model, showing that copula captures the dependence of real-world MMPP trace effectively, which in turn helps the prediction. In addition, copula-based predictions have other benefits compared to AR(1) model, as shown in the next section.

The Stability of Copula-based Model

Nowadays, a network flow may pass through many middleboxes, which may transform the traffic with some (potentially unknown) functions. In some scenarios, we may need to consider another counting process closely associated with the incoming traffic, *e.g.*, the number of CPU resources or the size of cache space that should be (dynamically)

allocated for processing the traffic. In these cases, the traffic is transformed with some functions or the new counting process can be viewed as the traffic transformed with a function. In the following, we study a new process $A'_i = \log(A_i)$ as an example. We note that the same conclusion could be drawn with other transformation functions. We call A'_i an *associated* trace.

With the invariant property of copulas, the temporal dependence between A'_i and A'_{i+1} in terms of copula remains the same as that between A_i and A_{i+1}. The measures ρ_τ, ρ_s and ρ_t among trace A'_i will also have the same theoretical results, since all of them could be derived with copula. However, since Pearson correlation does not satisfy the invariant property with the above transformation, ρ of A'_i is not theoretically tractable and thus needs to be calculated from empirical statistics. Table 5.3 shows the measures of trace A'_i. Comparing Table 5.1 and Table 5.3, we can see that copula-based dependencies are all the same while Pearson correlation varies, indicating that copula is much more stable than Pearson correlation.

Table 5.3: Dependence Measures of the Associated Trace from Theoretical Analysis and Empirical Analysis

	ρ_τ	ρ_s	$\rho_t^+\|_{u=0.99}$	$\rho_t^-\|_{u=0.01}$	ρ
Theoretical	0.4788	0.6150	0.4067	0.3359	—
Empirical	0.4212	0.5897	0.3935	0.3248	0.5916

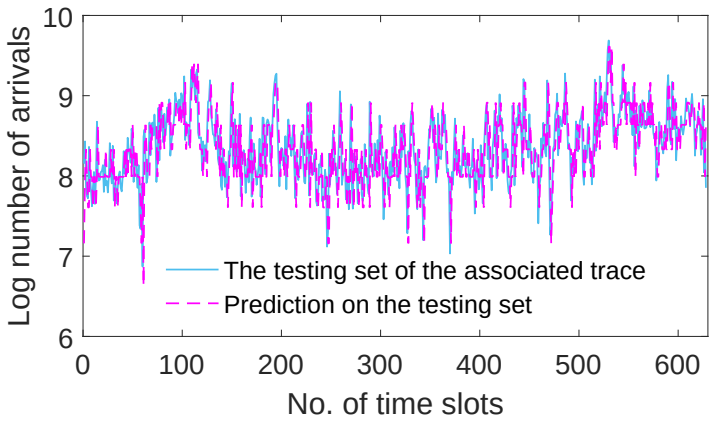


Figure 5.3: Prediction with theoretical copula on the testing set (last 20%) of the associated trace

Taking the advantage of invariant property, we do not need to rebuild the depen-

dence model when it comes to the prediction of A'_i with copula, because the same copula model for A_i can be applied and the marginal function of A'_i can be obtained from A_i by $M_{A'}(x) = M_A(2^x)$. Therefore, all copula models for A_i in Section 5.3.2 can be applied directly to predict A'_{i+1} given the A'_i value. Fig. 5.3 shows the prediction on the last 20% of the associated trace by using the same copula model of A_i. Nevertheless, without rebuilding the dependence model, the AR(1) and LPC(1) models for A_i applied to A'_i will lead to poor prediction performance. It is worth noting that rebuilding a new model for A'_i may be non-trivial due to the potentially unknown transformation function and the need of collecting and recording historical data of A'_i.

Table 5.4: One-Step Prediction RMSE on the Associated Trace with Different Training Percentages.

Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	0.4788	0.4344	0.5918	0.6953
60%	0.4653	0.4286	0.5868	0.6982
70%	0.4710	0.4393	0.5863	0.7137
80%	0.3955	0.3659	0.5471	0.6765
90%	0.3780	0.3655	0.5324	0.7390
aRMSE	0.4377	0.4068	0.5689	0.7045
IMP RATIO	37.87%	42.26%	19.26%	—

To test prediction performance without rebuilding a model, we apply the trained models (*i.e.*, copula, AR(1), and LPC(1)) from A_i to predict the associated trace A'_i. The one-step prediction RMSEs on A'_i of four methods are listed in Table 5.4. Both theoretical and parametric copulas outperform AR(1) and LPC(1) significantly. The results indicate that copula-based prediction is much more stable in the presence of traffic transformation, and both AR(1) and LPC(1) cannot capture the dependence in the associated trace accurately without a re-modeling process. Copulas take advantage of the invariant property to avoid the re-modeling process whenever an increasing functional transformation is imposed on the original traffic, leading to its much better performance over other models.

5.3.3 Case Study on HoMMPP Trace with Simulation

In real world, the availability of HoMMPP traffic traces is limited because it is not easy to identify them with proper fitting and goodness-testing methods. So we generate HoMMPP traces by simulation. We consider a scenario that there are 3 independent sources sending the traffic flows, with features similar to BCpAug89 trace, to one destination. The flow to the destination can be simulated as the aggregation trace of 3 independent MMPP traces generated by parameters $({}_AQ, {}_A\Lambda)$ shown in Eq.(5.9). The simulation lasts for 7200 seconds. We conduct the prediction on the generated HoMMPP trace. We analysed the trace in terms of arrival counts every second, *i.e.*, $\Delta = 1$. On the HoMMPP trace, we perform both one-step prediction and two-step prediction and compare copula models with others.

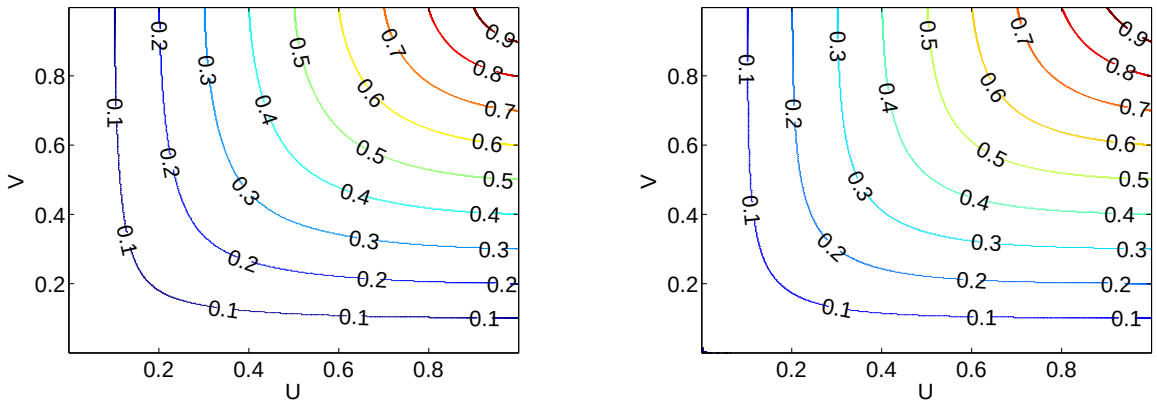
One-step Prediction on the HoMMPP Trace

When constructing HoMMPP copula matrix, the threshold is set as $\hat{a} = 2000$ according to observation of samples. Given parameters $({}_AQ, {}_A\Lambda)$ in Eq.(5.9), the theoretical copula of HoMMPP is calculated based on Theorems 16. The contour of the theoretical copula is shown in Fig. 5.4a. Note that even though we only compute the theoretical copula for $A_i^l \leq 2000$, it is almost the complete copula, because the threshold is large enough to make $C_{i'}^l(M^l(\hat{a}), M^l(\hat{a})) = 0.99998 \approx 1$ ($i' = 1, l = 3$ in this case), meaning that the probability for arrival counts to go beyond the threshold is extremely small. With the theoretical copula $C_{i'}^l$ constructed, dependence measures are calculated accordingly and compared with empirical results from trace data. Table 5.5 show the results, which indicate the accuracy of copula in modeling the trace. The theoretical Pearson correlation ρ is missing since its value on aggregate MMPP is extremely hard to calculate when the underlying MMPP has a large number of states.

Table 5.5: Dependence Measures of the HoMMPP trace from Theoretical Analysis and Empirical Analysis

	ρ_τ	ρ_s	$\rho_t^+\|_{u=0.99}$	$\rho_t^-\|_{u=0.01}$	ρ
Theoretical	0.5681	0.7329	0.3367	0.2484	—
Empirical	0.5500	0.7370	0.2857	0.2875	0.7566

Since the trace has similar upper and lower tail dependencies as shown in Table 5.5,



(a) Theoretical HoMMPP copula

(b) Parametric copula.

Figure 5.4: One-step copula contours for HoMMPP.

we choose Frank copula as the parametric copula. It indicates that the HoMMPP inherits the tail dependence features from single MMPP. The parameter of Frank copula is fitted according to different training set. Fig 5.4b shows the contour of the parametric copula trained from first 80% data of the HoMMPP trace. Contours in Fig. 5.4a and 5.4b are very similar. Their discrete L_2 norm distance [27] is 0.0100, which is small enough to justify the similarity of two copulas according to the results in [27].

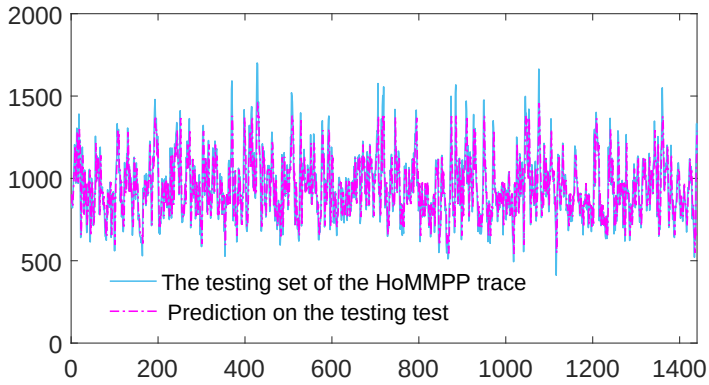


Figure 5.5: Prediction with theoretical HoMMPP copula on the testing set (last 20%)

With both theoretical copula and parametric copula, we perform one-step prediction on the HoMMPP trace. Fig. 5.5 shows the prediction with theoretical HoMMPP copula on the testing set of the last 20% arrival counts. We adjust the training percentage from 50% to 90%, and the prediction results are shown in Table 5.6. From the table, copula-based prediction has the highest IMP RATIO over the benchmark

Table 5.6: One-Step Prediction RMSE on the HoMMPP Trace with Different Training Percentage.

Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	133.2363	135.5447	138.2753	197.2492
60%	133.3298	137.0763	138.4576	199.7005
70%	131.5002	135.3443	136.6052	199.2604
80%	131.0944	132.6420	136.3443	198.3723
90%	127.9317	128.9178	133.2959	198.4436
aRMSE	131.4185	133.9050	136.5957	198.6052
IMP RATIO	33.82%	32.58%	31.22%	—

prediction regarding the aggregate MMPP traffic, indicating that copulas capture the temporal dependence of HoMMPP the best.

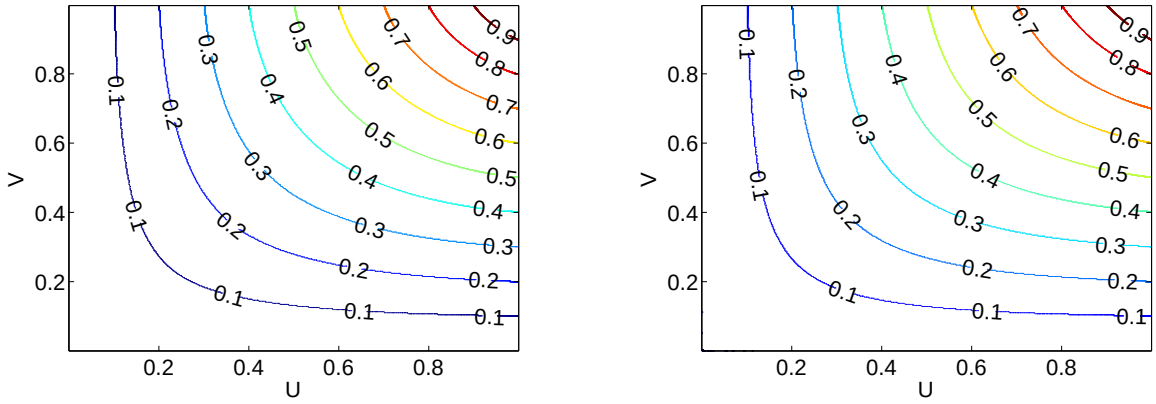
Two-step Prediction on the HoMMPP trace

We also experiment two-step prediction on the HoMMPP trace. That is, with any observation A_i^l in the test set, A_{i+2}^l is predicted. In order to make two-step prediction, the two-step theoretical HoMMPP copula is constructed as shown in Fig. 5.6a. Based on the two-step copula, the dependence measures are given and compared with empirical results in Table 5.7. Compared to one-step dependencies in Table 5.5, two-step dependencies between A_i^l and A_{i+2}^l are smaller because the dependence decreases as the step increases.

Table 5.7: Two-step Dependence Measures of the HoMMPP Trace from Theoretical Analysis and Empirical Analysis

	ρ_τ	ρ_s	$\rho_t^+\|_{u=0.99}$	$\rho_t^-\|_{u=0.01}$	ρ
Theoretical	0.2979	0.4189	0.1338	0.1077	—
Empirical	0.3372	0.4836	0.1690	0.1286	0.5104

Based on a training set of the HoMMPP trace, the parametric copula between A_i^l and A_{i+2}^l is trained accordingly. Fig. 5.6b shows the contour of the two-step



(a) Two-step theoretical HoMMPP copula

(b) Two-step Parametric copula for HoMMPP trace.

Figure 5.6: Two-step copula contours for HoMMPP.

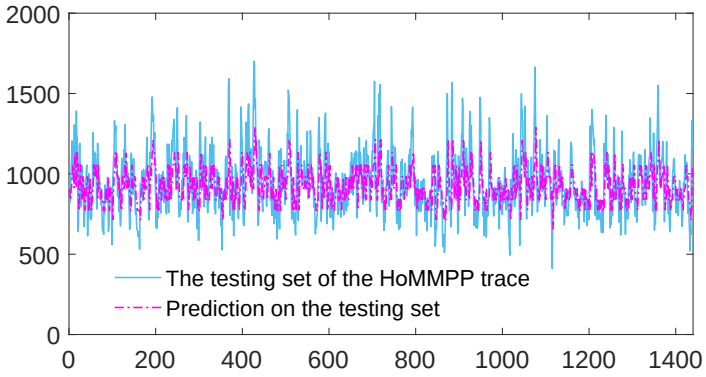


Figure 5.7: Two-step prediction with theoretical copula on the testing set (last 20%) of the HoMMPP trace

parametric copula trained from a training set consisting of the first 80% data of the HoMMPP trace. For two-step dependence, the theoretical copula and the parametric copula are also close to each other (Their discrete L_2 norm distance [27] is 0.0045).

With different training percentages, two-step predictions are performed on the HoMMPP trace. The prediction results of applying the theoretical copula on the last 20% of the HoMMPP trace are shown in Fig. 5.7. Prediction errors in terms of RMSE with different training percentages are shown in Table 5.8. Our copula models have significant improvement ratio (IMP RATIO) over benchmark model regarding the two-step predictions. Compared with AR(1), copulas also have a much better performance, indicating that copula can better characterize multi-step temporal dependence of arrival counts in MMPP.

Table 5.8: Two-Step Prediction RMSE on the HoMMPP Trace with Different Training Percentage.

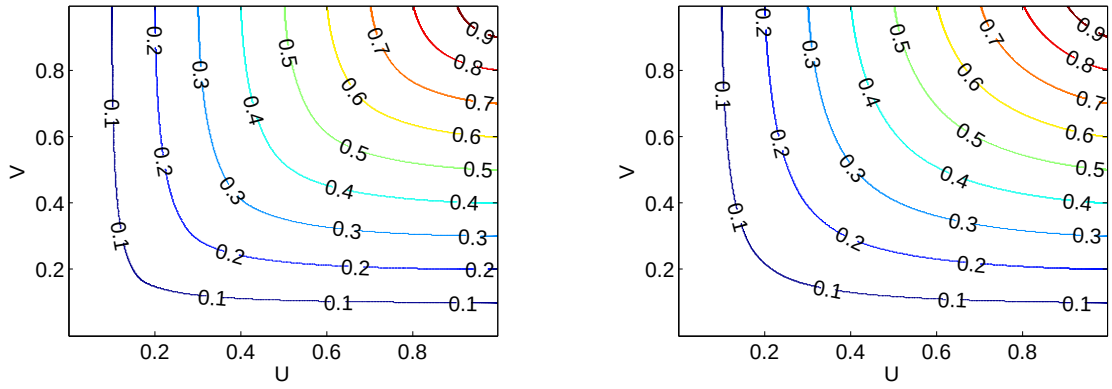
Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	173.3941	174.2526	196.5692	242.6005
60%	174.9574	176.4601	198.0623	247.9247
70%	173.3858	174.7150	197.3366	250.4306
80%	173.2769	174.3444	196.9888	250.7354
90%	169.6243	170.0910	194.5932	246.3572
aRMSE	172.9277	173.9726	196.7100	247.6097
IMP RATIO	30.16%	29.74%	20.56%	—

5.3.4 Case Study on HeMMPP trace

There are not many HeMMPP traces in real world ideal for the case study. Besides BCpAug89 trace, we add another Bellcore trace, BCpOct89 trace, for study. BCpOct89 trace record LAN traffic for about 1759.62 seconds. Analysing the traffic arrival in every 1 second, the BCpOct89 trace is fitted into a 13-state MMPP with parameters $({}_O Q, {}_O \Lambda)$ listed in Eq.(5.10). Since BCpAug89 trace and BCpOct89 traces are modelled by heterogeneous MMPPs, their aggregation, with them chopped into the same length, is ideal as a HeMMPP trace for prediction. Based on the observations of HeMMPP trace, the threshold for marginal and copula matrix computation is chosen as $\hat{a} = 1500$. The probability that the arrival count A_i^l exceeds the threshold is less than 0.01, resulting very few observations beyond the threshold.

$$
{}_{O}Q = \begin{pmatrix}
-1.00 & 0.75 & 0.00 & 0.25 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.04 & -0.64 & 0.26 & 0.25 & 0.06 & 0.02 & 0.00 & 0.02 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.13 & -0.72 & 0.34 & 0.16 & 0.03 & 0.03 & 0.02 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.01 & 0.06 & 0.12 & -0.68 & 0.31 & 0.13 & 0.04 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.10 & 0.25 & -0.74 & 0.20 & 0.11 & 0.06 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.04 & 0.09 & 0.23 & -0.71 & 0.20 & 0.10 & 0.03 & 0.02 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.03 & 0.06 & 0.31 & -0.68 & 0.16 & 0.08 & 0.02 & 0.01 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.01 & 0.02 & 0.04 & 0.19 & 0.34 & -0.81 & 0.16 & 0.05 & 0.01 & 0.01 & 0.01 \\
0.00 & 0.00 & 0.00 & 0.01 & 0.04 & 0.09 & 0.23 & 0.29 & -0.83 & 0.14 & 0.04 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.03 & 0.02 & 0.07 & 0.22 & 0.28 & -0.80 & 0.13 & 0.05 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.13 & 0.21 & 0.33 & -0.71 & 0.04 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.17 & 0.50 & 0.17 & -0.83 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & -1.00
\end{pmatrix},
$$

$$
{}_{O}\Lambda = (1125.89, 995.67, 873.46, 759.24, 653.02, 554.81, 464.59, 382.37, 308.15, 241.94, 183.72, 133.50, 91.28).
\tag{5.10}
$$



(a) One-step HeMMPP copula.

(b) Two-step HeMMPP copula.

Figure 5.8: Copula contours for HeMMPP.

The one-step and two-step HeMMPP copula are constructed as shown in Fig. 5.8. Based on copulas, we conduct one-step and two-step predictions on the HeMMPP trace. The training percentage is adjusted from 50% to 90%. The prediction accuracy is shown in Table 5.9 and 5.10. From the comparison, the copulas-based prediction has great improvement on accuracy over LPC(1) method. It also outperform AR(1) regarding the dependence modeling as well as the trace predictions.

Table 5.9: One-Step Prediction RMSE on the HeMMPP trace with Different Training Percentages.

Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	175.7469	172.1645	177.5212	206.1258
60%	179.5210	180.3826	181.8779	206.0690
70%	184.5205	185.6766	187.2252	209.7664
80%	185.9480	175.9524	188.0131	213.9332
90%	201.4238	186.3535	203.8665	226.9021
aRMSE	185.4320	180.1059	187.7008	212.5593
IMP RATIO	12.76%	15.27%	11.69%	—

Table 5.10: Two-Step Prediction RMSE on the HeMMPP trace with Different Training Percentages.

Training Percentage	Theoretical Copula	Parametric Copula	AR(1)	LPC(1)
50%	187.3233	191.8553	204.7920	215.7492
60%	187.0732	197.0069	205.4122	208.3386
70%	188.6817	197.3415	207.6971	217.6215
80%	187.6349	189.2811	207.2992	224.9221
90%	200.2666	196.8840	220.5304	250.7282
aRMSE	190.1959	194.4738	209.1463	223.4719
IMP RATIO	14.89%	12.98%	6.41%	—

5.4 Summary

Both real-world traffic trace and simulated trace are used to evaluate the copula model and its application to traffic prediction. In Section 5.3.2, the trace BCpAug89 is chosen to evaluate the accuracy of using MMPP copula to model the real-world trace. Prediction on a transformed trace from BCpAug89 in Section 5.3.2 shows that copula has much better dependence characterization and makes more accurate prediction than existing models in the presence of traffic transformation. Experiments in Sections 5.3.3 and 5.3.4 show copula's good performance on HoMMPP and HeMMPP, and multi-step dependence modeling.

From all the experiments, copula-based model has advantages over other dependence models (AR(1) and LPC(1)) in three aspects: First, it provides theoretical dependence structure of MMPP, including one or multiple step dependence of single MMPP and superposition of MMPPs. Second, it provides more information on dependence beyond linear scope. Third, it is more stable than other models. In the presence of traffic transformation, copula-based model does not require rebuilding a new dependence model but still guarantees accuracy.

Chapter 6

Application of MMPP Copulas in Composite Cloud Service Provisioning

In this chapter, we will use MMPP copulas in Chapter 4 and copula-based prediction in Chapter 5 to predict arrivals of cloud calls and design a dynamic service provisioning policy accordingly.

6.1 Introduction

Service composition has been broadly used to aggregate a set of services that work collaboratively to carry out a particular business task [45]. In the area of cloud computing, service composition has fostered a large service brokerage market, where cloud brokerages can deploy a new business service by integrating basic services from a pool of cloud service providers [45]. These basic services are normally offered with applications in virtual machines over cloud, and as such they are called virtualized functions (VFs) in the context.

A conceptual model of service composition in cloud computing is shown in Fig. 6.1. The conceptual model illustrates a large category of composite services. For example, the composite service could be the service that helps a customer with travel planning, where VFs consist of flight searching, hotel booking, tour recommendation, payment service, and so on. To distinguish service requests from end users to the cloud brokerage and the requests from cloud brokerage to VFs, we call the former as tasks and the

latter as calls, as shown in Fig. 6.1. As another example, Microsoft recently released Azure Service Fabric [56], with which the functional parts making up a service are split into small units that can be individually deployed, updated, distributed, and scaled. While the smaller units are run in containers rather than directly on VMs, Azure Service Fabric adopts a similar composite service model.

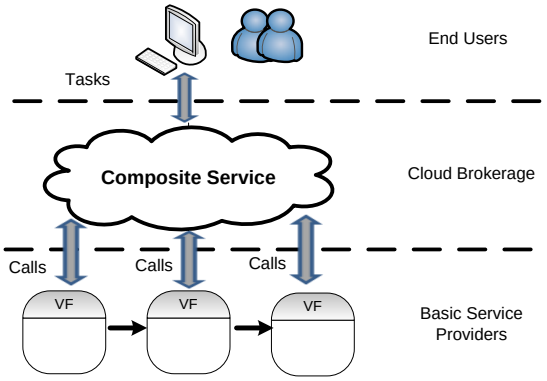


Figure 6.1: The conceptual diagram of service composition

It is critical that a composite service guarantees quality of service (QoS) to end users. QoS guarantee requires appropriate resource provisioning, and cloud computing provides us with an opportunity to dynamically scale up service capacity to alleviate the negative impact of burst service requests, and to scale down service capacity for cost saving. Nevertheless, QoS guarantee for composite services poses great challenges to the cloud brokerage due to the following reasons. First, existing auto-scaling techniques in commercial cloud, such as Amazon EC2, normally adjust the capacity of virtual machines (VMs) based on their utilization [1]. This auto-scaling strategy makes decisions only based on local VM utilization. Without a global view on the workflow of a composite service, existing auto-scaling techniques [1, 37, 74] may shift the bottleneck from one VF to another VF, leading to overall poor QoS in the presence of high task demands. Second, when the volume of task demands changes, without an accurate modeling or prediction of corresponding changes on the volume of calls to individual VFs, the auto-scaling of individual VFs may be triggered sequentially. This increases the delay of auto-scaling for the composite service.

A natural idea to tackle the above challenges is to orchestrate the auto-scaling of VFs based on a global view of composite service at the cloud brokerage. We call this idea *collaborative auto-scaling*, in the sense that VFs scale up/down their capacity cooperatively to maintain QoS of the composite service.

There are several key challenges in designing effective collaborative auto-scaling. First, how can we capture the dependence in the amount of calls to different VFs? While a good dependence model lays the foundation for collaborative auto-scaling, dependence modeling is difficult since a task to the composite service may trigger different amount of calls at different VFs. In other words, when the arrival rate of end users' tasks becomes high, the triggered amount of calls at different VFs may scale up differently. Second, with the dependence structure captured, how can we utilize the dependence to predict future service calls to different VFs? Third, how can we properly adjust the capacity of VFs to guarantee the QoS of a composite service, taking into consideration the delay in scaling up/down the capacity of VFs and tiered capacity levels in the cloud environment? This chapter makes use of MMPP copulas to address the above questions and propose a collaborative auto-scaling policy as well.

6.2 Related Work

Most Cloud providers, *e.g.*, Amazon, Windows Azure, and Google, offer rule based auto-scaling features to deal with time variant application workloads. Related work in designing auto-scaling mechanisms include workload forecasting, performance modeling, and cost optimal resource provisioning. In [74], a second order autoregressive moving average method (ARMA) is used to predict workload and an analysis algorithm for response time is used to find out the bottleneck server (with the highest utilization) under the predicted workload. In [33], a queuing network is used to model the relationship among response time, workload, and allocated resource. Kalman filter is used to derive the parameters used in the queuing network model. Cost optimal resource allocation in auto-scaling can be found in [81, 37, 55]. Systematic design on monitoring technique and scaling event handlers can be found in [84].

6.3 System Model

We first introduce our mathematical model to study the performance of composite services. Following the conceptual diagram in Fig. 6.1, we have the following assumptions:

1. We assume that the total number of VFs involved in the composite service is m. To fulfill a business *task* from end users, the composite service needs to make

a series of *calls* to VFs. The (call) arrivals to ordered sequence of VFs in the composite service is also referred as the workflow of the task. As an example shown in Fig. 6.2, the workflow of Task A is arrivals to VF1, arrivals to VF2, and arrivals to VF3.

2. We assume that the workflow may pass through m VFs in an arbitrary order and may pass a VF multiple times. The workflow may also skip a VF.

3. We assume that a task does not trigger parallel calls to VFs. Note that this assumption is needed to avoid the intricacy in modeling the degree of parallelism. This assumption, however, has no impact on the analytical results, because we can always decompose a task into sub-tasks such that each sub-task only makes sequential calls to VFs. In this case, the task is the aggregate of sub-tasks, each only making sequential calls to VFs.

4. We assume that when the task's workflow passes VF j each time, the calls to VF j follows a Poisson arrival process of mean rate λ_j. Note that this assumption is made not only because it eases analysis but also because Poisson arrivals have been used broadly as a good approximation for a variety of random arrivals [25].

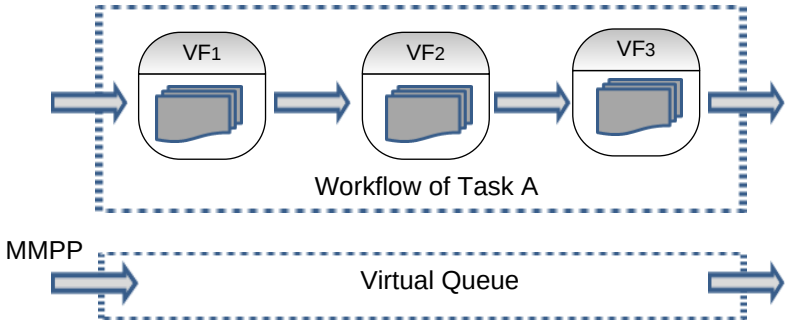


Figure 6.2: A queueing model for composite service

Performance modeling of composite services is difficult, because the workflows of different tasks may be different and a workflow may pass the same VF for multiple times. While network of queues is a natural choice for modeling the spatially separated queues, the non-deterministic order of queues and the multiple occurrences of the same queue in the queue chain make the analysis challenging.

Intuitively, there are some similarities between a composite service and the widely-used Markov Modulated Poisson Process (MMPP). MMPP assumes that a system

could be at different states and the arrivals to the system at different states may have different arrival rates. If we treat a VF as a state, then MMPP is a good model for a composite service. For a given task, only one VF, say VF j, works for the task at any given time instance, implying that the MMPP is at the state j if each VF corresponds to one state of the MMPP. With this intuition, we overcome the difficulty of modeling a composite service by approximating the workflow of a task as Markov Modulated Poisson Process (MMPP).

Remark 8. *Instead of focusing on the workflow of individual tasks, we study the long-term behavior of all tasks. Considering the aggregate workflows as a whole, it is not easy to model the arrival process as an m-state MMPP any more. Thus we will use copulas model the aggregate workflows.*

To summarize, the composite service is modeled as a single virtual queuing system with MMPP inputs, where each state of MMPP corresponds to an VF. In the sequel, we will answer the following critical questions in order to use the model for collaborative auto-scaling of VFs: (1) what is the dependence structure of calls in the composite service (Sections 6.4)? (2) how can we estimate the total resources of the virtual queue and accordingly decompose the total resource to that of individual VFs (Section 6.5)?

6.4 A Copula Model for Latent Dependence Structure in Service Composition

In the system model as we introduced in Section 6.3, the workflow of one task through the composite service can be approximated with single MMPP. In real-world scenario, the workflows to composite service systems will be from multiple tasks. Considering a system serving multiple tasks, the dependence structure among VFs in the composite service system will actually be presented by the temporal dependence in call arrivals from multiple tasks. As the system model is the same for all tasks, each task will follow the same MMPP, and the aggregate task workflow is a HoMMPP as defined in Chapter 4. HoMMPP is analysed in call arrival counts A_i^l within small time intervals Δ. In the application discussed in this chapter, A_i^l represents the number of call arrivals from l number of tasks to the system in the i-th time slot. With either theoretical copula or parametric copula, the temporal dependence between A_i^l

and $A^l_{i+i'}$ can be modeled. As the theorem and numerous experiments shown in Chapter 5, the value of $A^l_{i+i'}$ is effectively predicted by an observation of A^l_i. The predicted value shows the future call amount, and could be used to auto-scale the service that composite cloud system provides.

6.5 Collaborative Auto-Scaling of Virtualized Functions

6.5.1 Overview

Since the amount of calls in each time slot is considered as the aggregate of calls to VFs, this copula model between call arrival counts *implicitly* captures the dependence structure of VFs as well. With the help of the copula model, we introduce a strategy for collaboratively auto-scaling the capacity of VFs. The strategy also guarantees the utilization of each VF with individual auto-scaling embedded. Our method includes three main steps: (1) establishing the scaling matrix from the copula-based auto-scaling; (2) establishing the scaling matrix from the utilization-based individual auto-scaling; (3) collaboratively auto-scaling with the integrated scaling matrix. The scaling matrix is defined as the amount of capacity to scale up/down. Specifically, a positive value indicates scale-up and a negative value indicates scale-down.

To unify the measuring unit, both the workload and the capacity are measured in terms of rate. That is, the workload to the composite service and the capacity of a VF (or the virtual queue) are measured as the average rate (i.e., number of calls arrived/served per second). Before introducing the auto-scaling policy, we describe the following system parameters:

- Observation time interval Δ: As we defined in Section 6.4, we divide the time into time slots. Δ is the length of time slot. Each time slot is also the observation time interval, at the end of which scaling matrices are generated and decision of collaborative scaling is made.

- Scaling delay: A VF needs time to scale up/down. To align with our previous copula analysis, the scaling delay is measured in term of time slots, *i.e.*, the scaling delay is set to $d\Delta$.

- Capacity unit β: The capacity of VF is tiered at the multiples of β. This is because in practice people do not adjust the capacity of VF by an infinitesimal amount.
- Current capacity: γ is the current total capacity for all VFs, and γ_j is the capacity for VF j.

6.5.2 Copula-based Scaling Matrix

Copula-based scaling matrix considers the predicted workload. As we discussed in the above sections, the arrivals to the composite service system are considered as a HoMMPP. With copula modeling, the temporal dependence structure of the aggregate flows can be revealed and exploited to make inference on the future workload. Due to the scaling delay, we should construct the copula between A_i^l and A_{i+d+1}^l. Based on A_i^l sample value x_i, A_{i+d+1}^l is predicted by copula according to Theorem 17 or Theorem 20. The prediction of A_{i+d+1}^l is denoted as $\hat{x}$. The capacity needs to be adjusted to $\hat{x}/\Delta$ to satisfy the expected workload. Below we outline the auto-scaling procedure.

1. Scaling trigger: if $\hat{x}/\Delta - \gamma > m\beta$, the virtual capacity needs to scale up; if $\gamma - \hat{x}/\Delta > m\beta$, the virtual capacity needs to scale down.
2. Scaling dispatcher: if copula-based scaling is triggered, the predicted call arrival count is decomposed to VF j based on the stationary distribution of the CTMC associated to the modeling MMPP as: $\hat{x}(j) = \pi_j \hat{x}$. The capacity of VF j should be adjusted to the level of μ_j, such that $(\mu_j - 1)\beta < \hat{x}(j)/\Delta \leq \mu_j \beta$.

Overall, the copula-based scaling matrix for VF j is defined as

$$S_c = \begin{cases} \mu_j \beta - \gamma_j & \text{if } |\hat{x}/\Delta - \gamma| > m\beta \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

6.5.3 Utilization-based Individual Scaling Matrix

Utilization-based scaling is a traditional method for individual scaling [55]. The utilization-based individual scaling matrix is defined to guarantee that the utilization of each individual VF is not too high. This utilization-based scaling matrix carries information of whether the VF has a high amount of backlogs. This scaling matrix is also able to offset the prediction errors from the copula.

Given the utilization in the observation interval Δ for the VF j, ϱ_j, the utilization-based individual scaling matrix is defined as

$$S_u = \begin{cases} \beta & \text{if } \varrho_j \text{ is high, e.g., } > 0.9 \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

6.5.4 Integrated Scaling Matrix

For collaborative auto-scaling, we consider both workload information from S_c and the historical backlog information from S_u. Integrating the two matrices, the final collaborative scaling matrix S_g for a VF is calculated based on Table 6.1. The main idea is to scale up capacity when either the copula-based scaling matrix or utilization-based scaling matrix is positive, and to scale according to copula-based scaling matrix otherwise. This collaborative method can quickly modify the capacity following future workload trend without causing bottleneck or over provisioning in individual VFs.

Table 6.1: Calculation of Collaborative Scaling Matrix S_g

	$S_c > 0$	$S_c = 0$	$S_c < 0$
$S_u > 0$	$S_c + S_u$	S_u	S_u
$S_u = 0$	S_c	0	S_c

6.6 Performance Evaluation

To the best of our knowledge, there is no trace data for composite services currently available to the public. Due to this reason, we first study a real-world trace data for cloud requests, showing that MMPP indeed can be used to model the workflow of cloud requests. With the learned parameter from the real-world trace data, we then generate synthetic data with multiple workflows so that the performance of auto-scaling could be evaluated.

6.6.1 MMPP modeling of Real-world Cloud Trace

We first evaluate the effectiveness of MMPP modeling on Google cluster data [85], which is widely used for cloud computing performance analysis. Google cluster data

records arrival information to about 11,000 machines over a long period of 29 days in May 2011. The recorded data type related to our modeling are listed here:

- Time Stamp - arrival time in seconds of tasks since the start of data collection,
- TaskID - unique identifier of the executing task,
- JobID - unique identifier of the job to which the task belongs.

In our framework, calls are equivalent to the tasks in Google cluster data, tasks are equivalent to the jobs in Google cluster data. To align Google cluster data modeling with our previous analysis, the tasks in Google Cluster data is hereinafter called as calls, the jobs in Google cluster data called as tasks. Using this terminology, each task contains a series of calls to the Google cluster.

Recall that we model the workflow of a single task as MMPP, and the workflow is divided into small time slots for analysis. To match with the model and analysis, we set the length of time slot as $\Delta = 300$ seconds (5 minutes) and pre-process the Google cluster trace as follows:

1. count the number of call arrivals in every Δ seconds, denoted as $A_i(call)$,
2. count the number of task arrivals in every Δ seconds, denoted as $A_i(task)$,
3. normalize the call arrivals with the number of tasks, *i.e.*, $A_i = \frac{A_i(call)}{A_i(task)}$

The normalized call arrivals A_i could be regarded as the workflow of a single task to the Google cluster. We choose the normalized call arrivals A_i in the first 24 hours for modeling. The Google trace A_i is fitted into a MMPP model with the algorithm proposed in [39]. The learned MMPP is a 7-state MMPP with parameters (Q, Λ) as shown in Eq. (6.3). The unit of those parameters is second.

The common method to evaluate goodness of fitting real trace into MMPP is to compare a simulated trace from learned MMPP with the real trace statistically. Thus we generated a simulated trace for a duration of 24 hours according to the learned parameters (Q, Λ) in Eq.(6.3). The simulated arrivals are grouped into every 300 seconds, and then compared with Google trace in two statistical aspects - first two order of moments and distribution feature. The first two order of moments, including mean value, standard deviation (std), and skewness are compared in Table 6.2. The first two order of moments of the two traces are quite close, indicting that these two traces have similar statistical properties. Their distribution features are compared

$$
Q = \begin{pmatrix}
-0.0033 & 0 & 0 & 0.0008 & 0 & 0.0025 & 0 \\
0 & -0.0034 & 0.0008 & 0.0013 & 0.0013 & 0 & 0 \\
0 & 0.0002 & -0.0023 & 0.0014 & 0.0005 & 0.0002 & 0 \\
0.0001 & 0.0001 & 0.0002 & -0.0024 & 0.0014 & 0.0006 & 0 \\
0.0001 & 0.0001 & 0.0001 & 0.0007 & -0.0022 & 0.0012 & 0 \\
0 & 0.0001 & 0.0001 & 0.0004 & 0.0010 & -0.0016 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.0033 & -0.0033
\end{pmatrix},
$$

$$
\Lambda = \quad (0.7594, \quad 0.5715, \quad 0.4102, \quad 0.2756, \quad 0.1677, \quad 0.0865, \quad 0.0319). \tag{6.3}
$$

with Quantile-Quantile (Q-Q) plot in Fig 6.3. Both moments results and Q-Q plot figure shows that Google trace and the simulated trace have the same statistical behaviour and come from the same distribution.

Table 6.2: Comparison of The First Two Order of Moments of Arrival Counts in Every 300 Seconds

	Mean	Std	Skewness
Google trace	56.5882	37.9708	2.2016
Simulated trace	57.6021	37.5139	1.8323

6.6.2 Performance Evaluation with Synthetic Data

Our investigation on the Google trace data discloses that the workflow of a task submitted to cloud can be modeled with an MMPP model. Nevertheless, there is no trace data for composite services currently available to the public, and it is thus unclear how, and whether or not, the calls correspond to composite services. To overcome this problem, we evaluate the proposed collaborative auto-scaling method with synthetic data, created with simulation that aggregates multiple homogeneous workflows, each modeled with an MMPP with parameters learned from the Google trace as shown in Eq. (5.9). The *synthetic aggregate workflows* last for 48 hours and are equally split into two parts over time. The first half is used to train the copula parameter so as to model the temporal dependence of the aggregate homogeneous MMPPs. The second half is used as the input to a simulated composite service

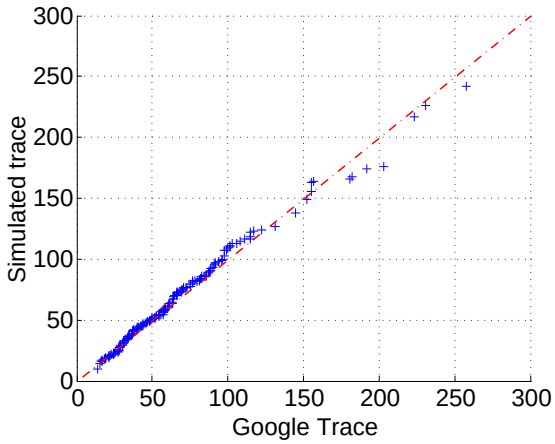


Figure 6.3: Q-Q plot of arrival counts in every 300 seconds

system with the implementation of our proposed collaborative auto-scaling policy. The parameters for simulated composite system are listed in Table 6.3.

Table 6.3: Parameters of Simulated Composite System

Simulation duration	24 hours
Number of workflows l	50
Observation time interval Δ	300 seconds
Scaling delay ($d = 1$)	300 seconds
Capacity unit β	0.01 per second

In order to implement and evaluate our solution to resource provisioning in cloud composite service system, we first model the temporal dependence of aggregate workflows (equivalently the dependence between VFs) with parametric copula as discussed in Section 4.5. With the parametric copula disclosing the dependence structure in composite services, the collaborative auto-scaling described in Section 6.5 is implemented. The proposed collaborative scaling will be compared with the traditional utilization-based individual scaling.

Copula modeling for Aggregate MMPPs

Considering the scaling delay, we need to construct the copula between A_i^l and A_{i+d+1}^l, where A_i^l represents the call arrival counts of l aggregate MMPP workflows in i-th time slot.

We use the mixture of Gumbel and Clayton copula to model temporal structure of HoMMPP:

$$\begin{aligned} C(u,v;\theta_1,\theta_2) =& 0.5 * \exp[-((-\log u)^{\theta_1} + (-\log v)^{\theta_1})^{1/\theta_1}] \\ &+ 0.5 * (u^{-\theta_2} + v^{-\theta_2} - 1). \end{aligned} \tag{6.4}$$

Gumbel copula is powerful in capturing upper tail dependence; Clayton copula, on the contrary, is used to model the lower tail dependence. This chosen parametric copula will be able to characterize the sudden increase and decrease in the MMPP workflows, and model the temporal dependence of MMPPs well. The first half of the synthetic aggregate workflows is fitted into the chosen copula to obtain the copula parameters as $\theta_1 = 1.4994, \theta_2 = 1.1654$.

Performance of collaborative auto-scaling

With the parametric copula built for HoMMPP, we can make inference on the arrival trend of the *synthetic aggregate workflows*. That is, given a observation of $A_i^l = x_i$, we predict the future call arrival count A_{i+d+1}^l. The inference on the second half of the *synthetic aggregate workflows* is shown in Fig. 6.4. The y-axis in the figure represents the number of aggregate arrivals within one time slot (5 minutes). From the figure, the predicted call arrival counts are close to the real call arrival counts. The *accuracy* of the prediction is also quantified by mean absolute percentage error (MAPE), defined as

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|\hat{x}_i - x_i|}{x_i}, \tag{6.5}$$

where $\hat{x}_i$ is the prediction for arrival count in i-th time slot and x_i is the real observed arrival count, n is the number of time slot in prediction period. The accuracy of copula-based inference is 0.0613, demonstrating the power of dependence structure modeled by copulas.

The collaborative auto-scaling is implemented following the policy in Section 6.5. We also implemented the traditional individual scaling algorithm for comparison. With individual scaling strategy, the capacity of each VF scales up β when its utilization is above 0.7, and scales down β when its utilization is below 0.2 [55]. We use the following performance matrices to compare the two auto-scaling strategies:

- Average response time of calls in seconds (ART): the total duration from the time when a call arrives to the time when the call departs over number of calls;

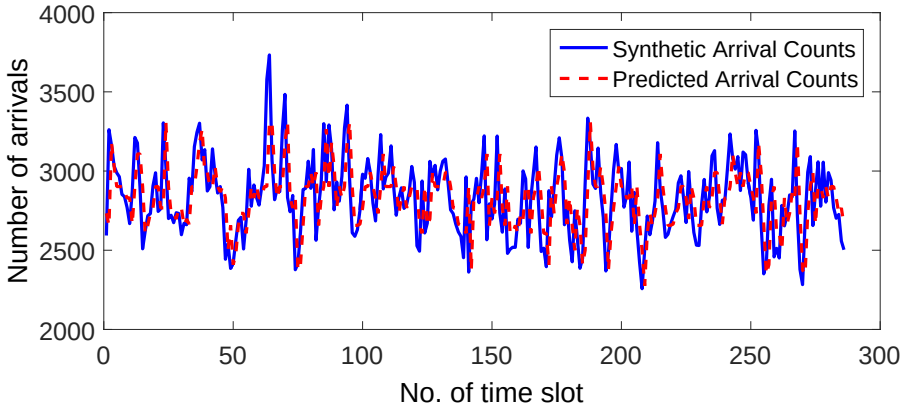


Figure 6.4: Copula-based inference on call arrival counts

- Average cost (AC): the total number of capacity units over number of time slots in the whole simulation duration.

Table 6.4: Simulation results with initial capacity as $\gamma_j = 1$

		VF1	VF2	VF3	VF4	VF5	VF6	VF7	Virtual Queue
Collaborative	ART	1067.1	546.8	776.6	126.1	51.5	50.3	320.6	252.5
Scaling	AC	64.9	93.2	122.2	309.6	307.2	175.6	2.2	1075.3
Individual	ART	160.4	72.1	110.1	13760.5	13952.2	1010.7	133.8	7809.6
Scaling	AC	108.9	140.7	167.9	243.5	243.5	215.0	19.3	1138.9

Table 6.5: Simulation results with initial capacity as $\gamma_j = 2$

		VF1	VF2	VF3	VF4	VF5	VF6	VF7	Virtual Queue
Collaborative	ART	1067.5	547.3	756.6	124.8	46.7	50.1	320.6	248.5
Scaling	AC	64.9	93.4	123.5	309.5	307.6	175.9	2.5	1077.6
Individual	ART	51.3	15.8	17.4	1234.1	1322.3	1.0	44.7	751.3
Scaling	AC	157.3	187.3	212.4	331.7	329.6	255.3	70.5	1544.5

The simulation results are shown in Table 6.4 and Table 6.5. Without using any prior-knowledge, we initialize the capacity of seven VFs equally. For experiment of Table 6.4, we choose small initial values, *i.e.*, $\gamma_j = 1 \quad (j = 1, \ldots, 7)$. For experiment of Table 6.5, we choose large initial values, *i.e.*, $\gamma_j = 2 \quad (j = 1, \ldots, 7)$. Using these two experiments, we investigate whether auto-scaling can adjust the capacity

following the actual workload quickly, and at the same time keep the response time and cost as small as possible. Table 6.4 and 6.5 record the performance matrices of each VF, as well as that of the virtual queue. Since virtual queue is an abstract concept for the integration of all the VFs, its performance matrices are, in fact, the performance matrices of the whole composite service system. From the level of virtual queue, we can observe that the collaborative auto-scaling performs better than individual scaling in the measure of both average response time and average cost. The results indicate that the copula modeling of dependence structure is effective. The collaborative auto-scaling makes good use of the prediction information from copulas to reduce the total cost while maintaining a small response time.

6.7 Summary

We have presented a new collaborative auto-scaling algorithm based on the temporal dependence of call arrives in cloud-based composite services. A key insight in our work is to model a task to the composite service as an MMPP. This, in turn, allows us to use copula analysis of MMPPs for understanding the dependence structure between calls to a composite service as well as predicting future calls. Our technical contributions include applying parametric copula models for incoming call prediction. Using real-world trace data and synthetic data, we have demonstrated that our collaborative auto-scaling method performs much better than the traditional auto-scaling method in which each VF auto-scales its capacity independently based on its local view of VF utilization.

Chapter 7

Application of MMPP Copulas in Parameter Estimation

In this chapter, we apply the theoretical copula in Chapter 4 to develop an accurate and fast parameter estimation method for MMPP.

7.1 Introduction

MMPP can capture a large range of traffic types, ranging from multimedia traffic, Poisson traffic, to burst traffic [31, 41, 79]. For all the applications of MMPP, the parameter estimation method is necessary for modeling.

The parameter estimation problem of MMPP has been studied for decades. Existing estimation methods can be broadly split into two categories. One category of work is maximum likelihood (ML) estimation with its implementation via expectation-maximization (EM) algorithm. Among the existing work, most research methodologies estimate MMPP parameters using data of inter-arrival times (or arrival times), and very few estimation methods can deal with data of number of arrivals over evenly-slotted time, which we call *arrival counts* in this thesis. In addition, none of the existing methods have utilized the *functional dependence* structure in MMPP traffic, that has the potential to further enhance the performance of parameter estimation.

In practice, there are some scenarios where arrival counts data is much more easier to capture and process. For instance, in Chapter 3, the aggregate of Skype flows is studied in terms of arrival counts in order to estimate its queueing performance. To give another example, arrival counts data is always used in performance monitoring

tools such as Windows Performance Monitor [63] to save memory resources especially for long-term recording. Since arrival counts are a more readily available form of data from most performance monitoring tools and, in addition, given the high cost of capturing and storing inter-arrival times, we are motivated to build an efficient estimation with only arrival counts data.

Nevertheless, the convenience of using arrival counts comes with a cost, since the arrival counts group arrivals within a time slot and thus contain less information than data of exact inter-arrival times. The loss of information makes estimation with arrival counts much more challenging than that with inter-arrival times. Up till now, only a few papers have proposed estimation methods that can learn MMPP parameters from arrival counts [6, 39, 13, 63]. Compared with the extensive studies of MMPP estimation based on inter-arrival times, MMPP estimation based on arrival counts is a relatively new topic and, for reasons described above, a much harder problem.

We tackle this challenging problem by utilizing the MMPP copula derived in Chapter 4. Traditional ML estimation emphasizes on likelihood, which is a variable representing joint behavior of the whole process, leading to high computational cost. In this chapter, we consider the joint behavior of successive arrival counts, *i.e.,* A_i and A_{i+1}. Taking advantage of the copula, the joint behavior of successive arrival counts can be modeled by studying marginal distribution of arrival counts and copula between A_i and A_{i+1} separately. Thus an estimation algorithm, termed as MarCpa, is proposed in this chapter to estimate MMPP parameters from arrival counts. MarCpa is fast and accurate, and it only includes two basic steps: one for marginal matching and one for copula matching.

7.2 Related Work

The parameter estimation problem of MMPP has been studied for several decades. According to different fitting objectives, the traditional estimation algorithms can be mainly categorized into two groups: the maximum likelihood estimation (MLE) algorithms and the moment-based algorithms. The former type was shown to achieve consistent results [75]. The MLE-based algorithms were implemented via expectation-maximization (EM) algorithm in [76]. Rydén's EM algorithm for estimating MMPP in [76] was further enhanced to ease the calculation of integrals [72, 28], and to estimate parameters from observations of either arrival times or arrival counts [13, 28].

The moment-based algorithms learn MMPP parameters by finding the moments, such as marginal moments and autocovariance. Compared with the MLE-based algorithms, the moment-based algorithms are usually fast and emphasize more on emulating specific dependence structures of real traces. For instance, moment-based algorithms were broadly used to emulate the self-similarity or long range dependence (LRD) of network traffic [2, 3, 87, 47, 77, 80]. The superposition of 2-state MMPPs was shown to be capable of modeling the self-similarity [2]. A high dimensional MMPP is constructed with superposition of 2-state MMPPs, because the moment of a 2-state MMPP is easy to compute. Following this idea, the superposition of 2-state MMPPs has been used to model the self-similarity or LRD of network traffic traces by matching their asymptotic covariances [3] or exact variances over different time scales [47, 77, 80, 87]. The learned MMPP parameters were integrated into queueing theory to predict the queueing performance.

In addition to the above two main categories, other fitting algorithms have also been developed. Algorithms were developed to fit IP traces into discrete MMPP by assuming that the Poisson arrivals of each state fall into certain range of variation [6, 39]. A Bayesian learning algorithm based on the posterior probability was developed to model and detect the bursty events [41]. The most recent algorithm learns MMPP parameters by first detecting the points of state switching and then estimating the arrival rates at the corresponding state [14].

Among all the literature in MMPP parameter estimation, most utilized the inter-arrival times, and only a few (e.g., [6, 39, 13, 63]) utilized arrival counts, which are related to our work of Chapter 7. Our work in Chapter 7 also uses arrival counts but differs significantly from the related works, since none of existing works used copula to analyse MMPP. We develop a two-step estimation method under this new analytical framework.

7.3 Copula-based Parameter Estimation of MMPP

With copula analysis of arrival counts in MMPP in Section 4.4.1, we develop an estimation method, called MarCpa, which consists of two matching steps: 1) matching theoretical marginal distribution of arrival counts with empirical marginal distribution from traces to learn the parameters $\Pi = (\pi_1, \cdots, \pi_m)$ and $\Lambda = (\lambda_1, \cdots, \lambda_m)$; 2) after Π and Λ are determined, matching theoretical copula into empirical copula from traces to determine the rest parameter Q. With the two steps, the proposed estimation

method will fully model the joint behavior of successive arrival counts. Moreover, our method with two separate matching steps will keep computational cost low. In the rest of this section, we will explain the proposed estimation method step by step.

7.3.1 Matching Marginal Distribution

The goal of this step is to match the empirical distribution of arrival counts of the sample trace with theoretical distribution. Given a sample trace $\{x_i\}_{1\leq i\leq n}$ with n number of arrival counts observed, the empirical distribution value is calculated as

$$\hat{u}_i \equiv \hat{M}(x_i) = \frac{1}{n}\sum_{i'=1}^{n} \mathbf{1}(x_{i'} \leq x_i), \quad \forall 1 \leq i \leq n. \tag{7.1}$$

The goal is to minimize the difference between the theoretical marginal distribution and the empirical marginal distribution, *i.e.*, to minimize the following objective function W_1

$$W_1 = \sum_{i=1}^{n} (u_i - \hat{u}_i)^2, \tag{7.2}$$

where $u_i = M(x_i)$ is calculated with Theorem 10.

The parameters involved in marginal distribution matching are Π and Λ. Considering that $\pi_1 + \pi_2 + \cdots + \pi_m = 1$, π_m can be always determined by $(\pi_1, \cdots, \pi_{m-1})$. Thus there are only $2m-1$ parameters to estimate in this step. These parameters are combined into one vector as $\Theta_1 = (\pi_1, \cdots, \pi_{m-1}, \lambda_1, \cdots, \lambda_m)$. The parameter estimation in this step turns out to be an optimization problem of the following form:

$$\begin{aligned} &\Theta_1 = \operatorname*{argmin}_{\Theta_1} W_1, \\ \text{subject to} \quad &\begin{cases} 0 \leq \pi_1, \cdots, \pi_{m-1} \leq 1, \\ 0 \leq \pi_1 + \cdots + \pi_{m-1} \leq 1, \\ \lambda_1, \cdots, \lambda_m \geq 0. \end{cases} \end{aligned} \tag{7.3}$$

The optimization in Eq. (7.3) is a constrained non-linear problem. The existing methods to directly deal with constrained non-linear optimization include geometric programming, quadratic programming, gradient-based methods, and metaheuristic methods such as genetic algorithm and simulated annealing [34]. Geometric programming and quadratic programming cannot be used here because the objective function

W_1 is a function of parameters Θ_1, which is much more complex than geometric or quadratic. Gradient-based method, which finds local optima, works efficiently in memory and computation. It can often find reasonably good solutions in a relatively short time. Because of this, it has been widely used to solve non-linear optimization in many applications [52]. Genetic algorithms and simulated annealing search for the global optima. As they iterate randomly, these two algorithms suffer from uncertain outcomes and may find solutions very slowly. Therefore, we use a gradient-based method, gradient descent, to solve the optimization in Eq. (7.3). The key steps of gradient descent method are parameter initialization, gradient derivation, choice of step size, and stopping criteria, which will be explained in detail below.

Parameter Initialization

The first step of our gradient descent method, is to initialize the values of parameters Λ and Π. Λ is initialized as local maxima on the frequency of observed arrival rate. Given a sample sequence of arrival counts $\{x_i\}_{1\leq i\leq n}$, the arrival rate sequence is $\{x_i/\Delta\}_{1\leq i\leq n}$. Detecting the local maxima on frequency of the arrival rate sequence helps to locate the most frequent but distinct arrival rates appearing in the sample, and these detected rates are reasonable initial values for Λ.

Fig. 7.1 shows an example where a 3-state MMPP is initialized with $\Lambda^{(0)} = (1, 8, 16)$ based on detection of local maxima on arrival rate frequency. The number of local maxima to detect can be set as known or unknown, which means we can either specify the number of states m or leave it to be automatically determined as the number of local maxima that the program could find. Thus the estimation method is flexible about the choice of number of states of MMPP.

To determine the stationary distribution Π, we need to first initialize the state in every time slot S_i. The initial value of the state in i-th time slot S_i is set as the state that has the closest arrival rate to observed arrival rate, *i.e.*,

$$S_i^{(0)} = \underset{j=1}{\overset{m}{\operatorname{argmin}}} |\lambda_j^{(0)} - x_i/\Delta|, \quad i = 1, 2, \cdots, n.$$

Based on initial values of $S_i^{(0)}$, the stationary distribution is initialized as

$$\pi_j^{(0)} = \frac{\mathbf{1}(S_i^{(0)} = j)}{n}, \quad j = 1, 2, \cdots, m-1.$$

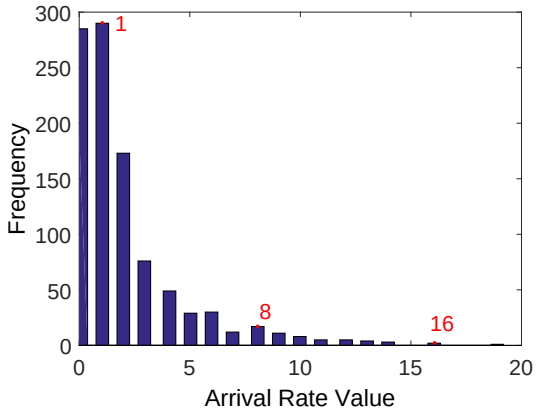


Figure 7.1: An example of the initialization of parameter Λ

Gradient of Parameters

The key step of gradient descent method is to obtain the gradient of parameters. In our problem, the parameter gradient $\frac{\partial W_1}{\partial \Theta_1}$ consists of $\frac{\partial W_1}{\partial \pi_j}$ for $j = 1, 2, \cdots, m-1$ and $\frac{\partial W_1}{\partial \lambda_j}$ for $j = 1, 2, \cdots, m$. The closed-forms for the two gradients are derived in Theorems 21 and 22:

Theorem 21. *The gradient of distribution probability π_j is*

$$\frac{\partial W_1}{\partial \pi_j} = \sum_{i=1}^{n} 2(u_i - \hat{u}_i)(G_j(x_i) - G_m(x_i)), \quad j = 1, \cdots, m-1. \tag{7.4}$$

Proof. Based on the marginal u_i given in Theorem 10 Chapter 4, the gradient is derived as follows:

$$\begin{aligned}
\frac{\partial W_1}{\partial \pi_j} &= \sum_{i=1}^{n} 2(u_i - \hat{u}_i)\frac{\partial u_i}{\partial \pi_j} \\
&= \sum_{i=1}^{n} 2(u_i - \hat{u}_i)\frac{\partial(\sum_{j'=1}^{m-1} \pi_{j'} G_{j'}(x_i) + (1 - \sum_{j'=1}^{m-1} \pi_{j'}) G_m(x_i))}{\partial \pi_j} \\
&= \sum_{i=1}^{n} 2(u_i - \hat{u}_i)\frac{\partial(G_m(x_i) + \sum_{j'=1}^{m-1} \pi_{j'}(G_{j'}(x_i) - G_m(x_i)))}{\partial \pi_j} \\
&= \sum_{i=1}^{n} 2(u_i - \hat{u}_i)(G_j(x_i) - G_m(x_i)).
\end{aligned}$$

□

Theorem 22. *The gradient of arrival rate λ_j is*

$$\frac{\partial W_1}{\partial \lambda_j} = \sum_{i=1}^{n} -2\pi_j \Delta (u_i - \hat{u}_i) e^{-\lambda_j \Delta} \frac{(\lambda_j \Delta)^{x_i}}{x_i!}, j = 1, \cdots, m. \tag{7.5}$$

Proof. Based on the marginal u_i given in Theorem 10 in Chapter 4,

$$\begin{aligned}\frac{\partial W_1}{\partial \lambda_j} &= \sum_{i=1}^{n} 2(u_i - \hat{u}_i) \frac{\partial u_i}{\partial \lambda_j} = \sum_{i=1}^{n} 2(u_i - \hat{u}_i) \frac{\partial (\sum_{j'=1}^{m} \pi_{j'} G_{j'}(x_i))}{\partial \lambda_j} \\ &= \sum_{i=1}^{n} 2(u_i - \hat{u}_i) \pi_j \frac{\partial G_j(x_i)}{\partial \lambda_j},\end{aligned}$$

where

$$\begin{aligned}&\frac{\partial G_j(x_i)}{\partial \lambda_j} = \frac{\partial (e^{-\lambda_j \Delta} \sum_{x=0}^{x=x_i} \frac{(\lambda_j \Delta)^x}{x!})}{\partial \lambda_j} = \frac{\partial (e^{-\lambda_j \Delta})}{\partial \lambda_j} + \sum_{x=1}^{x_i} \frac{\partial (\frac{e^{-\lambda_j \Delta} (\lambda_j \Delta)^x}{x!})}{\partial \lambda_j} \\ &= -\Delta e^{-\lambda_j \Delta} + \Delta \sum_{x=1}^{x_i} \frac{\partial (\frac{e^{-\lambda_j \Delta} (\lambda_j \Delta)^x}{x!})}{\partial (\lambda_j \Delta)} = -\Delta e^{-\lambda_j \Delta} - \Delta \sum_{x=1}^{x_i} \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^x}{x!} + \Delta \sum_{x=1}^{x_i} \frac{e^{(-\lambda_j \Delta)} x (\lambda_j \Delta)^{x-1}}{x!} \\ &= -\Delta e^{-\lambda_j \Delta} - \Delta \sum_{x=1}^{x_i} \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^x}{x!} + \Delta \sum_{x=1}^{x_i} \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^{x-1}}{(x-1)!} \\ &= -\Delta e^{-\lambda_j \Delta} - \Delta \sum_{x=1}^{x_i} \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^x}{x!} + \Delta \sum_{x=0}^{x_i - 1} \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^x}{x!} \\ &= -\Delta e^{-\lambda_j \Delta} - \Delta \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^{x_i}}{x_i!} + \Delta e^{-\lambda_j \Delta} \\ &= -\Delta \frac{e^{(-\lambda_j \Delta)} (\lambda_j \Delta)^{x_i}}{x_i!}.\end{aligned}$$

□

With the gradients derived in Theorems 21 and 22, the parameters in each iterative

step is updated to $\Theta_1^{(r+1)} = \Theta_1^{(r)} - \alpha^{(r)} \frac{\partial W_1}{\partial \Theta_1}|_{\Theta_1=\Theta_1^{(r)}}$, where the specific updates are

$$\pi_j^{(r+1)} = \pi_j^{(r)} - \alpha^{(r)} \frac{\partial W_1}{\partial \pi_j}|_{\Theta_1=\Theta_1^{(r)}} \qquad j = 1, 2, \cdots, m-1;$$
$$\lambda_j^{(r+1)} = \lambda_j^{(r)} - \alpha^{(r)} \frac{\partial W_1}{\partial \lambda_j}|_{\Theta_1=\Theta_1^{(r)}} \qquad j = 1, 2, \cdots, m.$$

Note that $\pi_m^{(r+1)}$ is always determined by $\pi_m^{(r+1)} = 1 - \pi_1^{(r+1)} - \cdots - \pi_{m-1}^{(r+1)}$. In above updates, $\alpha^{(r)}$ is the step-size of r-th iterative step. The choice of step-size is discussed next.

Choice of Step-size

The step-size $\alpha^{(r)}$ is a positive value, which can be changed as iteration number r increases. For the optimization problem in Eq. (7.3), since we are required to consider the constraints on parameters, the step-size will be adjusted accordingly to guarantee that the constraints are satisfied. As the initial parameters $\Theta_1^{(0)}$ certainly satisfy the constraints, we only need to guarantee that every $\Theta_1^{(r+1)}$ obtained from $\Theta_1^{(r)}$ satisfies the constraints. Specifically, the step-size of r-th iteration $\alpha^{(r)}$ is randomly chosen as a positive number satisfying the constraints:

$$\begin{cases} 0 \le \pi_j^{(r)} - \alpha^{(r)} \frac{\partial W_1}{\partial \pi_j}|_{\Theta_1=\Theta_1^{(r)}} \le 1, \quad j = 1, 2, \cdots, m-1; \\ 0 \le \sum_{j=1}^{m-1} \pi_j^{(r)} - \alpha^{(r)} \sum_{j=1}^{m-1} \frac{\partial W_1}{\partial \pi_j}|_{\Theta_1=\Theta_1^{(r)}} \le 1; \\ \lambda_j^{(r)} - \alpha^{(r)} \frac{\partial W_1}{\partial \lambda_j}|_{\Theta_1=\Theta_1^{(r)}} \ge 0, \quad j = 1, 2, \cdots, m. \end{cases} \tag{7.6}$$

Stopping Criteria

The iteration continues until it meets some predetermined criteria. Two stopping criteria are considered in our gradient descent progress: 1) the iteration r reaches a predetermined maximum number of iteration n_Itr; 2) the decreasing ratio of the objective function W_1 drops below a preset threshold th, *i.e.*, $\frac{W_1^{(r-1)} - W_1^{(r)}}{W_1^{(r-1)}} \le th$. Whenever any of the two stopping criteria is satisfied, the iteration stops and values of parameters Π and Λ are returned as output.

7.3.2 Matching Copula

In the second step, the theoretical copula in Theorem 11 in Chapter 4 is matched to the empirical copula calculated from the trace. Given a sample trace of arrival counts $\{x_i\}_{1\leq i\leq n}$, empirical copula value of successive arrival counts is

$$\hat{\xi}_i = \frac{1}{n-1}\sum_{i'=1}^{n-1}\mathbf{1}(x_{i'} \leq x_i, x_{i'+1} \leq x_{i+1}), \quad \forall 1 \leq i \leq n-1. \tag{7.7}$$

The goal of the matching is to minimize the difference between theoretical copula of successive arrival counts and their empirical copula as represented by W_2:

$$W_2 = \sum_{i=1}^{n-1}(\xi_i - \hat{\xi}_i)^2, \tag{7.8}$$

where ξ_i is calculated from theoretical copula given in Theorem 11, *i.e.*, $\xi_i = C_1(M(x_i), M(x_{i+1}))$. With the parameters Π and Λ determined, the parameters required to estimate ξ_i are entries of matrix $P(\Delta) = [p(\Delta)]_{m\times m}$. Thus, we obtain the following optimization problem:

$$\begin{aligned} P(\Delta) &= \operatorname*{argmin}_{P(\Delta)} W_2, \\ \text{subject to} \quad &\begin{cases} \Pi P(\Delta) = \Pi, \\ \sum_{j_2=1}^{m} p_{j_1 j_2}(\Delta) = 1, \quad j_1 = 1, 2, \cdots, m \\ p_{j_1 j_2}(\Delta) \geq 0. \end{cases} \end{aligned} \tag{7.9}$$

The optimization problem in Eq. (7.9) is a classical quadratic programming problem with linear constraints. To make it clearer, we now define several vectors and matrices to illustrate how the problem will be solved:

- **Parameter vector Θ_2**
  Θ_2 is a $m^2 \times 1$ parameter vector reshaped from $P(\Delta)$ in the way:

$$\begin{aligned} \Theta_2 = (&p_{11}(\Delta), p_{21}(\Delta), \cdots, p_{m1}(\Delta), p_{12}(\Delta), p_{22}(\Delta), \cdots \\ &, p_{m2}(\Delta), \cdots, p_{1m}(\Delta), p_{2m}(\Delta), \cdots, p_{mm}(\Delta))^T, \end{aligned}$$

  i.e., the k-th element in Θ_2 is $p_{j_1 j_2}(\Delta)$ where $j_1 = (k-1)\%m + 1$, $j_2 = [(k-1)/m] + 1$, $\%$ is modulo operation and $[\cdot]$ operation rounds down values to

integers.

- **Coefficient matrix** H
  H is a $(n-1) \times m^2$ dimensional matrix with its elements as

$$h_{ik} = G_{j_2}(x_{i+1})G_{j_1}(x_i)\pi_{j_1}$$
$$\text{where } j_1 = (k-1)\%m + 1 \text{ and } j_2 = [(k-1)/m] + 1.$$

  Based on Theorem 11, $\xi_i = h_i * \Theta_2$, where h_i is the i-th row vector of H. Moreover, we have

$$\begin{pmatrix} \xi_1 \\ \vdots \\ \xi_{n-1} \end{pmatrix} = H\Theta_2.$$

- **Constraints coefficient matrix** E
  E is a $2m \times m^2$ matrix with all non-zero elements defined as

$$E_{ik} = \pi_j \text{ for}$$
$$i = 1, \cdots, m, j = 1, \cdots, m \text{ and } k = (i-1)m + j,$$
$$E_{ik} = 1 \text{ for}$$
$$i = m+1, \cdots, 2m, j = 1, \cdots, m \text{ and } k = (j-1)m + i.$$

- **Constraints vector** b
  The vector b is a $2m \times 1$ vector defined as $b = (\pi_1, \pi_2, \cdots, \pi_m, 1, 1, \cdots, 1)^T$.

Example 4. *Taking a 2-state MMPP as an example, the four vectors or matrices defined above are in the following forms:*

$$\Theta_2 = (p_{11}(\Delta), p_{21}(\Delta), p_{12}(\Delta), p_{22}(\Delta))^T,$$

$$H = \begin{pmatrix} \pi_1 G_1(x_1)G_1(x_2) & \pi_2 G_2(x_1)G_1(x_2) & \pi_1 G_1(x_1)G_2(x_2) & \pi_2 G_2(x_1)G_2(x_2) \\ \pi_1 G_1(x_2)G_1(x_3) & \pi_2 G_2(x_2)G_1(x_3) & \pi_1 G_1(x_2)G_2(x_3) & \pi_2 G_2(x_2)G_2(x_3) \\ \vdots & \vdots & \vdots & \vdots \\ \pi_1 G_1(x_{n-1})G_1(x_n) & \pi_2 G_2(x_{n-1})G_1(x_n) & \pi_1 G_1(x_{n-1})G_2(x_n) & \pi_2 G_2(x_{n-1})G_2(x_n) \end{pmatrix},$$

$$E = \begin{pmatrix} \pi_1 & \pi_2 & 0 & 0 \\ 0 & 0 & \pi_1 & \pi_2 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$
$$b = (\pi_1, \pi_2, 1, 1)^T.$$

With Θ_2, H, E, b defined, Θ_2 fully represents $P(\Delta)$, H helps derive objective function W_2 in terms of Θ_2, and E and b characterize the constraints on Θ_2. The optimization problem in Eq. (7.9) is reformulated as

$$\begin{aligned} \Theta_2 &= \underset{\Theta_2}{\operatorname{argmin}} W_2 = \underset{\Theta_2}{\operatorname{argmin}} \sum_{i=1}^{n-1} (\xi_i^2 - 2\hat{\xi}_i \xi_i) \\ &= \underset{\Theta_2}{\operatorname{argmin}} \begin{pmatrix} \xi_1 \cdots \xi_{n-1} \end{pmatrix} \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_{n-1} \end{pmatrix} - 2 \begin{pmatrix} \hat{\xi}_1 \cdots \hat{\xi}_{n-1} \end{pmatrix} \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_{n-1} \end{pmatrix} \\ &= \underset{\Theta_2}{\operatorname{argmin}} \frac{1}{2} \Theta_2^T H^T H \Theta_2 - \begin{pmatrix} \hat{\xi}_1 \cdots \hat{\xi}_{n-1} \end{pmatrix} H \Theta_2 \\ &\text{subject to} \quad \begin{cases} E\Theta_2 = b, \\ \Theta_2 \geq \mathbf{0}. \end{cases} \end{aligned} \tag{7.10}$$

Now the problem in Eq. (7.10) becomes clear as a classic quadratic programming with linear constraints. We thus use **quadprog**, a solver from Matlab optimization toolbox, to get the optimal values in Θ_2. According to the mapping rule between elements' indexes, $P(\Delta)$ is easily to obtain from Θ_2.

The final step to complete parameter evaluation is to recover the rate matrix Q from transition probability matrix $P(\Delta)$. As Δ is small, infinitesimal term $o(\Delta)$ is ignorable. Based on Eq.(4.1), Q can be approximated from $P(\Delta)$ with Eq. (7.11).

$$\begin{aligned} q_{j_1 j_2} &= (p_{j_1 j_2}(\Delta) - 1)/\Delta, \quad j_1 = j_2; \\ q_{j_1 j_2} &= p_{j_1 j_2}(\Delta)/\Delta, \quad j_1 \neq j_2. \end{aligned} \tag{7.11}$$

7.3.3 A Summary of MarCpa Algorithm

Matching marginal distributions in Section 7.3.1 and matching copula in Section 7.3.2 are combined to make our proposed MMPP parameter estimation algorithm, Mar-

Cpa algorithm, the sketch of which is shown in Algorithm 4. The step of matching marginal distributions solves a constrained non-linear optimization problem with gradient descent, with the time complexity $O(m \times n \times n\_\text{Itr})$. The step of matching copula solves a quadratic program with linear constraints. Since we use Matlab solver quadprog in this step, its time complexity depends on how Matlab implements its solver. Although MarCpa uses existing algorithms, gradient descent and Matlab solver, it is the first time that the estimation process is decomposed into separate steps with copula to ease analysis.

Algorithm 4 MarCpa Algorithm

Require: a sequence of arrival counts $\{x_i\}$, the length of time slots Δ
Ensure: MMPP parameters Λ and Q

// **First step: matching marginal distributions**
// Note that n_Itr and th are maximum iteration number and threshold value defined in Section 7.3.1
Determine initial parameters $\Lambda^{(0)}$ and $\Pi^{(0)}$ according to Section 7.3.1, and compute initial objective function $W_1^{(0)}$;
Initialize $\Lambda = \Lambda^{(0)}$, $\Pi = \Pi^{(0)}$, $W_1 = W_1^{(0)}$;
for $r \leftarrow 0 : n\_\text{Itr} - 1$ **do**
    Choose a proper step-size $\alpha^{(r)}$ according to Section 7.3.1;
    Update parameters $\Pi^{(r)}$ to $\Pi^{(r+1)}$, $\Lambda^{(r)}$ to $\Lambda^{(r+1)}$ based on Section 7.3.1, and compute objective function $W_1^{(r+1)}$;
    if $W_1^{(r+1)} < W_1$ **then**
        $\Lambda = \Lambda^{(r+1)}$, $\Pi = \Pi^{(r+1)}$, $W_1 = W_1^{(r+1)}$;
    end if
    if $(W_1^{(r)} - W_1^{(r+1)})/W_1^{(r)} \leq th$ **then**
        Break;
    end if
end for

// **Second step: matching copulas**
Construct matrices H and E, vector b based on their definitions in Section 7.3.2;
Obtain the optimal value of Θ_2 by inputting H, $\{\hat{\xi}_i\}_{1 \leq i \leq n-1}$, E, b, and $\mathbf{0}$ in a proper form to qradprog solver;
Reshape Θ_2 to $P(\Delta)$;
Recover parameter Q from $P(\Delta)$ according to Eq.(7.11).

7.4 Performance Evaluation

In this section, the performance of MarCpa is evaluated with a large number of simulations. We first use one simulated sample trace as a concrete example that presents ground truth along with the estimated parameters. The comparison of estimated parameters with ground truth parameters illustrates how well MarCpa retrieves parameters from arrival counts. As a further step, the evaluation is conducted on multiple simulations. The average goodness-of-fitting of multiple independent simulations quantifies the performance of MarCpa over different parameter settings. We also compare the performance of MarCpa with that of existing Expectation-Maximization (EM) learning algorithm (e.g., the one in [63]) and non-EM algorithm learning algorithm (e.g., the one in [39]).

7.4.1 Performance Evaluation Based on Ground Truth

We consider a 2-state MMPP with parameters

$$Q = \begin{pmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{pmatrix} = \begin{pmatrix} -1 & 1 \\ 0.1 & -0.1 \end{pmatrix},$$

$$\Lambda = (\lambda_1, \lambda_2) = (10, 1).$$

A trace was generated with simulation according to the above parameters for a period of 1000 unit of time. We group the arrivals within every 1 unit of time, *i.e.*, $\Delta = 1$. The arrival counts of this trace are shown in Fig. 7.2.

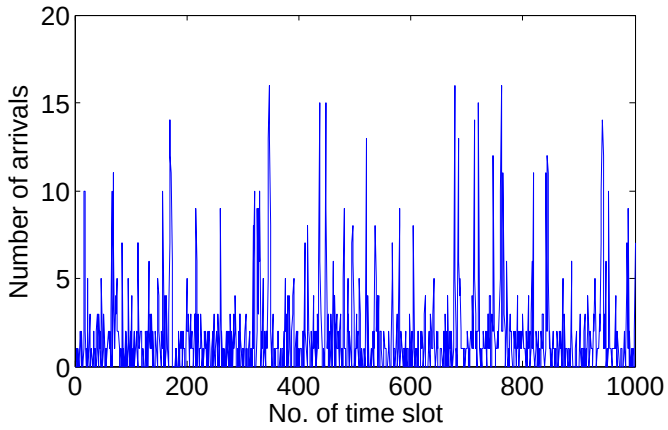


Figure 7.2: Arrival counts of simulation trace.

Table 7.1: Estimated parameters for the simulation trace.

	q_{11}	q_{22}	λ_1	λ_2
MarCpa	-1.0000	-0.0834	10.0000	1.0400
EM	-1.0650	-0.1070	10.4320	0.9508
non-EM	-0.5896	-0.0925	9.7500	1.2614
Ground truth	-1	-0.1	10	1

The estimated parameters with MarCpa are shown in Table 7.1. The table also contains the results from the Expectation-Maximization (EM) learning algorithm in [63] and those from the non-EM algorithm learning algorithm proposed in [39]. Among the three results, the estimated parameters from MarCpa look closer to the ground truth parameters. To quantitatively compare the three estimation methods and check their results with ground truth, Kolmogorov-Smirnov (K-S) tests are performed. Essentially, the K-S test compares a fitted distribution with a sample in terms of cumulative distribution function. We extend the classic K-S test to measure the difference of copulas so the temporal dependence goodness-of-fitting is evaluated as well. We use the following two distances for measurement:

$$D_M = \max_{i=1}^{n} |u_i - \hat{u}_i|, \tag{7.12}$$

$$D_C = \max_{i=1}^{n-1} |\xi_i - \hat{\xi}_i|. \tag{7.13}$$

The critical value of K-S test $D_{0.01}$ is determined by the size of samples, n. $D_{0.01}$ for D_M is calculated as $D_{0.01} = 1.63/\sqrt{n}$ and for D_C is $D_{0.01} = 1.63/\sqrt{n-1}$. If the sample statistics D_M and D_C are both equal to or smaller than the corresponding critical value, the sample trace is accepted as one from estimated model; otherwise, it is rejected as sample from estimated model. The K-S test results with parameters estimated from three different methods plus ground truth parameters are listed in Table 7.2. Compared to the two state-of-art estimation methods, our proposed MarCpa method has the closest values of D_M and D_C to ground truth parameters, indicating its estimation result is the closest to the ground truth. Moreover, MarCpa is the only method that retrieves parameters passing the K-S test, implying that MarCpa is the most effective algorithm to identify hidden MMPP from the trace. Therefore, we conclude that the MMPP model estimated with our proposed MarCpa algorithm

performs the best to recover ground truth and characterize the sample trace.

Table 7.2: Kolmogorov-Smirnov test results on sample trace.

	D_M	$D_{0.01}$	D_C	$D_{0.01}$
MarCpa	**0.04317**	0.05153	**0.03975**	0.05157
EM	0.05254		0.05834	
non-EM	0.09414		0.10163	
Ground truth	**0.03734**		**0.04260**	

The running time of MarCpa, EM and non-EM algorithms are recorded in Table 7.3. All the three estimation algorithms run in a computer with Intel Core i7-2600 CPU @ 3.40GHz, 4.00 GB RAM. Among the three, the non-EM algorithm is fastest, and the EM algorithm takes the longest time. While MarCpa is slightly slower than the non-EM algorithm, MarCpa returns much more accurate results.

Table 7.3: Running time in seconds.

MarCpa	EM	non-EM
0.9305	2.0900	0.6119

7.4.2 Performance Evaluation Based on Average Goodness-of-Fitting and Running Time

In this section, we evaluate the effectiveness and stability of MarCpa method by analysing its average performance results over multiple independent experiments. We first consider a 3-state MMPP with following parameters:

$$Q = \begin{pmatrix} -1 & 0.5 & 0.5 \\ 0.25 & -0.50 & 0.25 \\ 0.05 & 0.05 & -0.1 \end{pmatrix},$$

$$\Lambda = (10, 5, 1).$$

Thirty independent traces are generated from this 3-state MMPP: ten of them are traces with a duration of 1000 units of time, ten are traces with a duration of

5000 units of time, and the rest ten are with a duration of 10000 units of time. The arrivals counts are the number of arrivals in every 1 unit of time, *i.e.*, $\Delta = 1$.

For each trace, the three methods, MarCpa, EM and non-EM, are applied to learn MMPP parameters. The performance evaluation includes both goodness-of-fitting and running time. The goodness-of-fitting is measured by K-S distances D_M and D_C. The running times are converted with $\log_{10}$ operation for easy illustration.

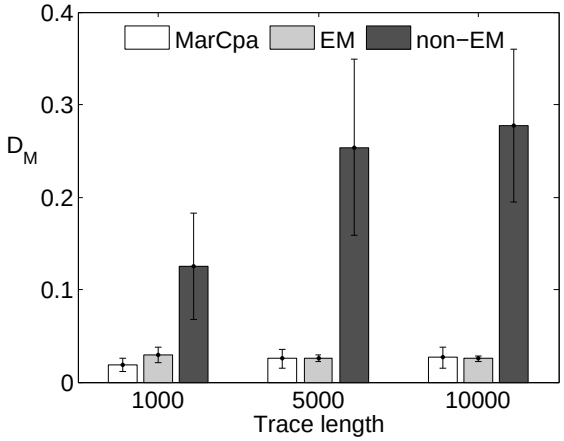


Figure 7.3: Performance in D_M for 3-state MMPP traces.

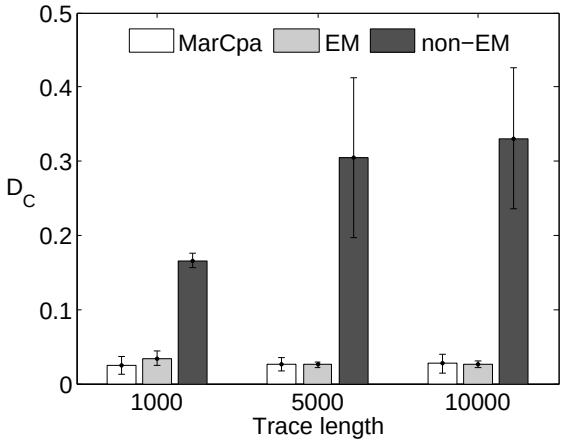


Figure 7.4: Performance in D_C for 3-state MMPP traces.

Fig. 7.3, Fig. 7.4 and Fig. 7.5 show the average performance results of the three methods in D_M, D_C, and running time, respectively. In these figures, the results are grouped by trace length. Ten results from traces with the same length are analysed in their average and standard deviation, that is, each bar shows the average estimation performance on ten traces with length indicated by horizontal axis, and the error

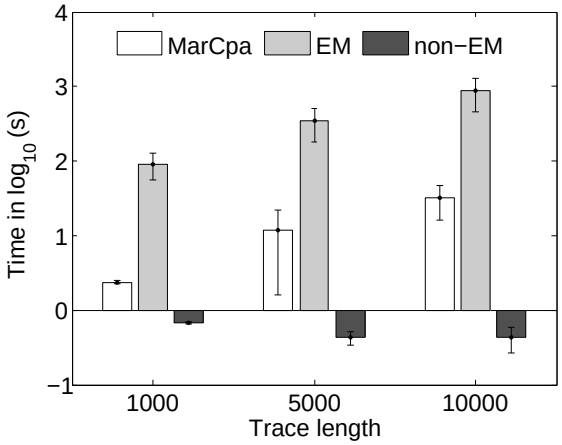


Figure 7.5: Performance in running time for 3-state MMPP traces.

bars represent the variation (variation in $\log_{10}$ for running time) of ten independent experiments.

The above experiments and evaluation are repeated on a 5-state MMPP with parameters:

$$Q = \begin{pmatrix} -1 & 0.25 & 0.25 & 0.25 & 0.25 \\ 0.2 & -0.8 & 0.2 & 0.2 & 0.2 \\ 0.125 & 0.125 & -0.5 & 0.125 & 0.125 \\ 0.075 & 0.075 & 0.075 & -0.3 & 0.075 \\ 0.025 & 0.025 & 0.025 & 0.025 & -0.1 \end{pmatrix},$$

$$\Lambda = (20, 15, 10, 5, 1).$$

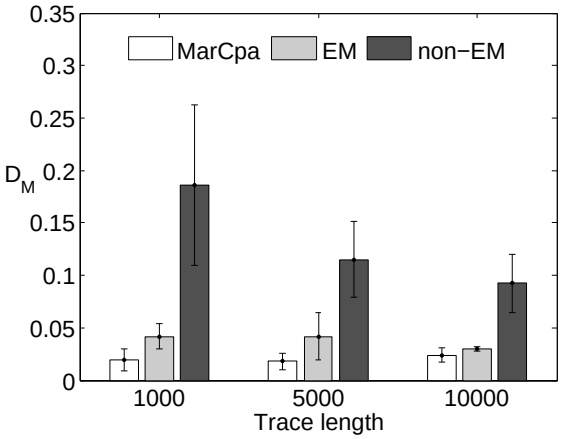


Figure 7.6: Performance in D_M for 5-state MMPP traces.

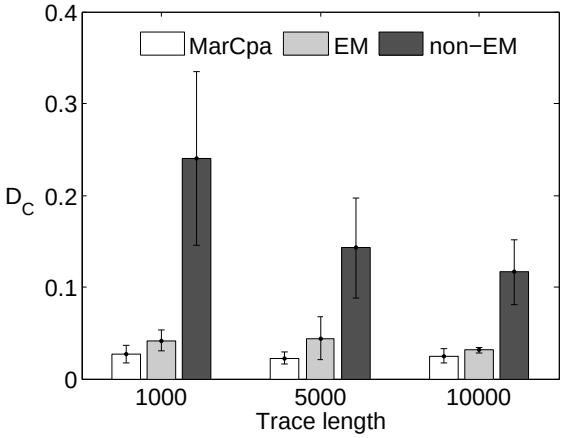


Figure 7.7: Performance in D_C for 5-state MMPP traces.

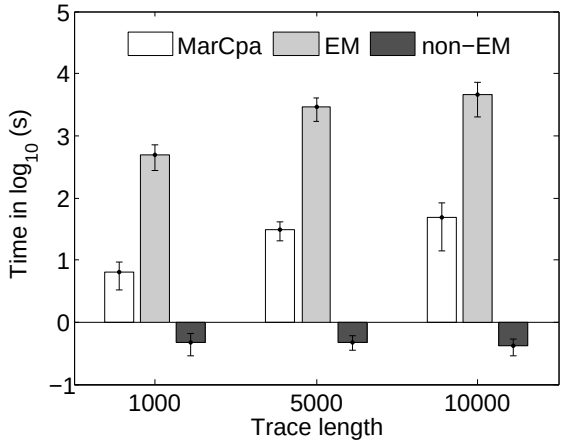


Figure 7.8: Performance in running time for 5-state MMPP traces.

The estimation performances in D_M, D_C, and running time of the three methods on the 5-state MMPP are compared in Fig 7.6, Fig. 7.7 and Fig. 7.8, respectively.

Experiments on both the 3-state MMPP and the 5-state MMPP show that the proposed MarCpa algorithm has a stable and effective performance. Compared with the EM algorithm, MarCpa has competitive (even better in the 5-state MMPP) goodness-of-fitting results but uses over 10 times less running time. Compared with the non-EM algorithm, it takes a longer time but has much better fitting. Table 7.4 shows the number of times that the estimated parameters pass K-S tests. Among the 30 independent experiments of either 3-state or 5-state MMPP, MarCpa is the best at retrieving parameters that represent the MMPP trace well.

To conclude, the proposed MarCpa algorithm improves the learning accuracy over both EM and non-EM algorithms, and improves the time-efficiency significantly over

Table 7.4: Ratio of experiments that pass K-S tests.

	MarCpa	EM	non-EM
3-state MMPP experiments	14/30	13/30	0/30
5-state MMPP experiments	17/30	10/30	0/30

the EM algorithm.

7.5 Summary

This chapter proposed a new learning algorithm, called MarCpa, to estimate MMPP parameters from arrival counts data. With copula theory, it is the first time that functional dependence has been applied to estimate MMPP parameters. With extensive simulation evaluation, our proposed method outperforms existing methods by improving estimation accuracy and by keeping running time small.

Chapter 8

Conclusions and Future Work

In this thesis, we investigate copula theory, and apply copula theory to analyse the contemporaneous dependence between traffic flows and the temporal dependence in one traffic flow. Our analytical results are applied in several application scenarios in computer networks. In this chapter, we summarize the contributions of this thesis and discuss future research directions.

8.1 Contemporaneous Dependence Modeling

In Chapter 3, we apply copula to model the contemporaneous dependence between traffic flows. With a case study of Skype traffic flows, we show how to model the contemporaneous dependence between network flows with copula, and how copula disclose the dependence between flows in a novel way. With copula analysis, we obtain tight and accurate models for aggregate flows, which further benefits statistical network calculus by tightening the performance bounds of network backlog and queueing delay.

As the first work to explore copula analysis in stochastic network calculus, it is expected to motivate a new spectrum of interests in extending SNC research and further enhance its impact in practice. Along these lines, many interesting research problems deserve further investigation. These include copula structures with different sub-sampling techniques other than sliding windows, better dynamic scheduling and multiplexing strategies aligning with the underlying changes of traffic flows, and new types of copula structures tailored for specific network environment.

8.2 Temporal Dependence Modeling

In Chapter 4, we apply copula to model the temporal dependence in a traffic flow, which is modeled as Markov Modulated Poisson process. We model the temporal dependence of MMPP with copula by deriving the theoretical copula between arrival counts in different time slots. Recursive algorithms are developed to compute the theoretical copula of superposition of multiple independence MMPPs. We also propose the parametric copula modeling steps to model the temporal dependence of MMPP.

In Chapter 5, the temporal dependence of MMPP is applied for traffic prediction. With numerous case studies, we show that copula-based dependence works effectively to predict future arrivals of single MMPP flow, and future arrivals of superposition of homogeneous/ heterogeneous MMPP flows. The accuracy and stable traffic prediction demonstrates the power of copula modeling of temporal dependence.

In Chapter 6, we combine the contribution of the MMPP copula in Chapter 4 and copula-based prediction in Chapter 5, and design a service provisioning policy based on prediction of cloud future call arrivals. We study the call arrivals to composite cloud service system approximated as MMPP. With the copula modeling temporal dependence between call arrival counts and prediction made by copula, we can predict future service demand. A collaborative auto-scaling policy is proposed to fulfill future service demand and keep the cost low at the same time. With simulations, we show that our collaborative auto-scaling policy based on temporal dependence modeling outperforms traditional auto-scaling policy in which each component of composite cloud system scales capacity independently.

Another application of MMPP copula is investigated in Chapter 7. The temporal dependence in terms of copula is applied to estimate parameters of MMPP. On the basis of analytical results of marginal distribution and copula of MMPP, we propose a two-step matching algorithm to learn MMPP parameters from arrival counts. With extensive evaluations, our proposed estimation method works better than the state-of-art methods in the sense that it improves estimation accuracy and keeps running time small.

8.3 Future Work

In the future, the results of this thesis can be extended in several directions.

First, we can use copula-based temporal dependence to solve various challenges

in network domains. In our thesis, we mainly capture the temporal dependence in a specific traffic model, MMPP. However, copula, has the potential to characterize the temporal dependence of different types of traffic. Studying the temporal dependence of different traffic types will benefit applications involving traffic model.

Second, another future work direction is to apply high order copula to model temporal dependence for network traffic. This thesis uses 2-copula to model the dependence between arrival counts in two time slots. Using high order copula, the dependence among traffic from multiple time slots can be modeled and would be more powerful and general for many applications.

Third, we can explore the power of copula models in other network applications. For instance, when network is under attacks, the temporal dependence of traffic may change. Modeling temporal dependence with copula, the abnormal traffic could be differentiated from normal traffic. Thus copula analysis will help to identify different traffic as well as detect network anomaly.

Bibliography

[1] Amazon. Auto scaling. *http://aws.amazon.com/autoscaling/*, Accessed in July 2015.

[2] Allan T Andersen and Bo Friis Nielsen. An application of superpositions of two state markovian source to the modelling of self-similar behaviour. In *Proceedings of INFOCOM'97*, pages 196–204, Kobe, Japan, 1997. IEEE.

[3] Allan T Andersen and Bo Friis Nielsen. A markovian approach for modeling packet traffic with long-range dependence. *IEEE Journal on Selected Areas in Communications*, 16(5):719–732, 1998.

[4] Tomasz Andrysiak and Łukasz Saganowski. Network anomaly detection based on statistical models with long-memory dependence. In *Theory and Engineering of Complex Systems and Dependability*, pages 1–10. Springer, 2015.

[5] Kazim Azam and Michael K Pitt. Bayesian inference for a semi-parametric copula-based Markov chain, 2014. Working paper.

[6] Soshant Bali and Victor S Frost. An algorithm for fitting MMPP to IP traffic traces. *IEEE Communications Letters*, 11(2):207–209, 2007.

[7] Jerry Banks, John S Carson, Barry L Nelson, and David Nicol. *Discrete event system simulation*. Prentice hall, 2010.

[8] Aaron K Baughman, Richard Bogdany, Benjie Harrison, Brian OConnell, Herbie Pearthree, Brandon Frankel, Cameron McAvoy, Sandy Sun, and Clay Upton. IBM predicts cloud computing demand for sports tournaments. *Interfaces*, 46(1):33–48, 2016.

[9] Michael A Beck, Sebastian A Henningsen, Simon B Birnbach, and Jens B Schmitt. Towards a statistical network calculus–dealing with uncertainty in arrivals. In *INFOCOM, 2014 Proceedings IEEE*, pages 2382–2390. IEEE, 2014.

[10] Khalid Begain, Gunter Bolch, and Helmut Herold. *Practical performance modeling: application of the MOSEL language*. Springer Science & Business Media, US, 2012.

[11] Vladislav Bína and Radim Jiroušek. A short note on multivariate dependence modeling. *Kybernetika*, 49(3):420–432, 2013.

[12] Eric Bouyé, Valdo Durrleman, Ashkan Nikeghbali, Gaël Riboulet, and Thierry Roncalli. Copulas for finance-a reading guide and some applications. *Available at SSRN 1032533*, 2000.

[13] Lothar Breuer and Alfred Kume. An EM algorithm for markovian arrival processes observed at discrete times. In *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, pages 242–258. Springer, Berlin Heidelberg, 2010.

[14] Yulia Burkatovskaya, Tatiana Kabanova, and Sergey Vorobeychikov. CUSUM algorithms for parameter estimation in queueing systems with jump intensity of the arrival process. In *Information Technologies and Mathematical Modelling-Queueing Theory and Applications*, pages 275–288. Springer, Switzerland, 2015.

[15] Cheng-Shang Chang. *Performance guarantees in communication networks*. Springer, 2000.

[16] Tiberiu Chis and Peter G Harrison. Adapting hidden Markov models for online learning. *Electronic Notes in Theoretical Computer Science*, 318:109–127, 2015.

[17] Doo Il Choi, Tae-Sung Kim, and Sangmin Lee. Analysis of an MMPP/G/1/K queue with queue length dependent arrival rates, and its application to preventive congestion control in telecommunication networks. *European Journal of Operational Research*, 187(2):652–659, 2008.

[18] Florin Ciucu, Almut Burchard, and Jörg Liebeherr. A network service curve approach for the stochastic analysis of networks. In *ACM SIGMETRICS Performance Evaluation Review*, volume 33, pages 279–290. ACM, 2005.

[19] Florin Ciucu, Felix Poloczek, and Jens Schmitt. Sharp bounds in stochastic network calculus. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, pages 367–368. ACM, 2013.

[20] Florin Ciucu and Jens Schmitt. Perspectives on network calculus: no free lunch, but still good value. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 311–322. ACM, 2012.

[21] Mark E Crovella and Azer Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997.

[22] Rene L Cruz. A calculus for network delay. I. network elements in isolation. *Information Theory, IEEE Transactions on*, 37(1):114–131, Jan 1991.

[23] Rene L Cruz. A calculus for network delay. II. network analysis. *Information Theory, IEEE Transactions on*, 37(1):132–141, Jan 1991.

[24] Tibor Csóka and Jaroslav Polec. Modeling Poisson error process on wireless channels. *International Journal of Communication Networks and Information Security*, 7(1):1–7, 2015.

[25] Anirban DasGupta. Poisson processes and applications. In *Probability for Statistics and Machine Learning*, pages 437–462. Springer, 2011.

[26] Jadran Dobrić and Friedrich Schmid. Testing goodness of fit for parametric families of copulasapplication to financial data. *Communications in Statistics–Simulation and Computation*, 34(4):1053–1068, 2005.

[27] Valdo Durrleman, Ashkan Nikeghbali, Thierry Roncalli, et al. Which copula is the right one, 2000. Working paper.

[28] Robert J Elliott and W Paul Malcolm. Discrete-time expectation maximization algorithms for Markov-modulated Poisson processes. *IEEE Transactions on Automatic Control*, 53(1):247–256, 2008.

[29] Paul Embrechts, Filip Lindskog, and Alexander McNeil. Modelling dependence with copulas. *Rapport technique, Département de mathématiques, Institut Fédéral de Technologie de Zurich, Zurich*, 2001.

[30] Markus Fidler and Jens B Schmitt. On the way to a distributed systems calculus: An end-to-end network calculus with data scaling. In *ACM SIGMETRICS Performance Evaluation Review*, volume 34, pages 287–298. ACM, 2006.

[31] Wolfgang Fischer and Kathleen Meier-Hellstern. The Markov-modulated Poisson process (MMPP) cookbook. *Performance Evaluation*, 18(2):149–171, 1993.

[32] Henry J Fowler and Will E Leland. Local area network characteristics, with implications for broadband network congestion management. *IEEE Journal on Selected Areas in Communications*, 9(7):1139–1149, 1991.

[33] Anshul Gandhi, Parijat Dube, Alexei Karve, Andrzej Kochut, and Li Zhang. Adaptive, model-driven autoscaling for cloud applications. In *Proceedings of ICAC 14*, pages 57–64. USENIX Association, 2014.

[34] Amir Hossein Gandomi, Xin-She Yang, Siamak Talatahari, and Amir Hossein Alavi. *Metaheuristic applications in structures and infrastructures*. Newnes, 2013.

[35] Christian Genest and Anne-Catherine Favre. Everything you always wanted to know about copula modeling but were afraid to ask. *Journal of hydrologic engineering*, 12(4):347–368, 2007.

[36] Christian Genest, Bruno Rémillard, and David Beaudoin. Goodness-of-fit tests for copulas: A review and a power study. *Insurance: Mathematics and economics*, 44(2):199–213, 2009.

[37] Hamoun Ghanbari, Bradley Simmons, Marin Litoiu, Cornel Barna, and Gabriel Iszlai. Optimal autoscaling in an IaaS cloud. In *Proceedings of ICAC '12*, pages 173–178. ACM, 2012.

[38] Mahmood Mollaei Gharehajlu, Saadan Zokaei, and Yousef Darmani. Statistical analysis of different traffic types effect on QoS of wireless ad hoc networks. *Journal of Information Systems & Telecommunication*, 3(1(9)):7–15, 2015.

[39] Daniel P Heyman and David Lucantoni. Modeling multiple IP traffic streams with rate limits. *IEEE/ACM Transactions on Networking*, 11(6):948–958, 2003.

[40] Ling Hu. Dependence patterns across financial markets: a mixed copula approach. *Applied Financial Economics*, 16(10):717–729, 2006.

[41] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Learning to detect events with Markov-modulated Poisson processes. *ACM Transactions on Knowledge Discovery from Data*, 1(3):13, 2007.

[42] Y.M. Jiang. Network calculus and queueing theory: Two sides of one coin. In *Proceedings of VALUETOOLS 2009*, Pisa, Italy, Oct. 2009.

[43] Yuming Jiang. Stochastic network calculus for performance analysis of Internet networks–an overview and outlook. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*, pages 638–644. IEEE, 2012.

[44] Yuming Jiang and Yong Liu. *Stochastic network calculus*. Springer, 2008.

[45] Amin Jula, Elankovan Sundararajan, and Zalinda Othman. Cloud computing service composition: A systematic literature review. *Expert Systems with Applications*, 41(8):3809–3824, 2014.

[46] Thomas Karagiannis, Mart Molle, and Michalis Faloutsos. Long-range dependence ten years of Internet traffic modeling. *IEEE internet computing*, 8(5):57–64, 2004.

[47] Shoji Kasahara. Internet traffic modeling: Markovian approach to self-similar traffic and prediction of loss probability for finite queues. *IEICE Transactions on Communications*, 84(8):2134–2141, 2001.

[48] Pradeeban Kathiravelu and Luis Veiga. An expressive simulator for dynamic network flows. In *Cloud Engineering (IC2E), 2015 IEEE International Conference on*, pages 311–316. IEEE, 2015.

[49] Krishna H Koirala, Ashok K Mishra, Jeremy M D'Antoni, and Joey E Mehlhorn. Energy prices and agricultural commodity prices: Testing correlation using copulas method. *Energy*, 81:430–436, 2015.

[50] Krishna H Koirala, Ashok K Mishra, Joey Mehlhorn, et al. Using copula to test dependency between energy and agricultural commodities. In *2014 Annual Meeting, July 27-29, 2014, Minneapolis, Minnesota*. Agricultural and Applied Economics Association, 2014.

[51] Jean-Yves Le Boudec and Patrick Thiran. *Network calculus: a theory of deterministic queuing systems for the Internet*. Springer, 2001.

[52] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[53] Ian WC Lee and Abraham O Fapojuwo. Stochastic processes for computer network traffic modeling. *Computer Communications*, 29(1):1–23, 2005.

[54] Chengzhi Li, Almut Burchard, and Jörg Liebeherr. A network calculus with effective bandwidth. *IEEE/ACM Transactions on Networking*, 15(6):1442–1453, 2007.

[55] Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *Proceedings of SC 2011*, pages 1–12. ACM, 2011.

[56] Microsoft. Azure service fabric. *http://azure.microsoft.com/en-us/campaigns/service-fabric//*, Accessed in July 2015.

[57] Andrew W Moore and Konstantina Papagiannaki. Toward the accurate identification of network applications. In *International Workshop on Passive and Active Network Measurement*, pages 41–54. Springer, 2005.

[58] Luca Muscariello, Marco Mellia, Michela Meo, M Ajmone Marsan, and R Lo Cigno. Markov models of Internet traffic and a new hierarchical MMPP model. *Computer Communications*, 28(16):1835–1851, 2005.

[59] Roger B. Nelson. *An introduction to copulas*. Springer, New York, 2006.

[60] David Neuhäuser, Christian Hirsch, Catherine Gloaguen, and Volker Schmidt. A parametric copula approach for modelling shortest-path trees in telecommunication networks. In *Analytical and Stochastic Modeling Techniques and Applications*, pages 324–336. Springer, 2013.

[61] Marcel F Neuts. *Structured stochastic matrices of M/G/1 type and their applications*. Taylor & Francis, New York, USA, 1989.

[62] António Nogueira, Paulo Salvador, Rui Valadas, and António Pacheco. Modeling self-similar traffic through Markov modulated Poisson processes over multiple time scales. In *High-Speed Networks and Multimedia Communications*, pages 550–560. Springer, Berlin Heidelberg, 2003.

[63] Hiroyuki Okamura, Tadashi Dohi, and Kishor S Trivedi. Markovian arrival process parameter estimation with group data. *IEEE/ACM Transactions on Networking*, 17(4):1326–1339, 2009.

[64] Sergio Pacheco-Sanchez, Giuliano Casale, Bryan Scotney, Sally McClean, Gerard Parr, and Stephen Dawson. Markovian workload characterization for QoS prediction in the cloud. In *2011 IEEE International Conference on CLOUD*, pages 147–154, Washington, D.C., USA, 2011. IEEE.

[65] Andrew Patton. Copula methods for forecasting multivariate time series. *Handbook of Economic Forecasting*, 2:899–960, 2012.

[66] Andrew J Patton. Copula–based models for financial time series. In *Handbook of Financial Time Series*, pages 767–785. Springer, 2009.

[67] Andrew John Patton. *Applications of copula theory in financial econometrics*. PhD thesis, University of California, San Diego, 2002.

[68] Felix Poloczek and Florin Ciucu. A martingale-envelope and applications. *ACM SIGMETRICS Performance Evaluation Review*, 41(3):43–45, 2014.

[69] Ali Rajabi and Johnny W Wong. MMPP characterization of web application traffic. In *2012 IEEE 20th International Symposium on MASCOTS*, pages 107–114, Washington, D.C., USA, 2012. IEEE.

[70] Ali Rajabi and Johnny W Wong. Provisioning of computing resources for web applications under time-varying traffic. In *2014 IEEE 22nd International Symposium on MASCOTS*, pages 152–157, Paris, France, 2014. IEEE.

[71] Bruno Rémillard, Nicolas Papageorgiou, and Frédéric Soustra. Copula-based semiparametric models for multivariate time series. *Journal of Multivariate Analysis*, 110:30–42, 2012.

[72] William JJ Roberts, Yariv Ephraim, and Elvis Dieguez. On Rydén's EM algorithm for estimating MMPPs. *IEEE Signal Processing Letters,*, 13(6):373–376, 2006.

[73] Sheldon M. Ross. *Introduction to probability models*. Academic Press, Burlington, 2003.

[74] Nilabja Roy, Abhishek Dubey, and Aniruddha Gokhale. Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Proceedings of CLOUD*, pages 500–507. IEEE, 2011.

[75] Tobias Rydén. Parameter estimation for Markov modulated Poisson processes. *Stochastic Models*, 10(4):795–829, 1994.

[76] Tobias Rydén. An EM algorithm for estimation in Markov-modulated Poisson processes. *Computational Statistics & Data Analysis*, 21(4):431–447, 1996.

[77] Paulo Salvador, Rui Valadas, and António Pacheco. Multiscale fitting procedure using Markov modulated Poisson processes. *Telecommunication Systems*, 23(1-2):123–148, 2003.

[78] Mischa Schwartz. *Broadband integrated networks*. Prentice Hall PTR New Jersey, 1996.

[79] Steven L Scott. Detecting network intrusion using a Markov modulated nonhomogeneous Poisson process. *Available online*, 2001.

[80] Shou-Kuo Shao, Malla Reddy Perati, Meng-Guang Tsai, Hen-Wai Tsao, and Jingshown Wu. Generalized variance-based markovian fitting for self-similar traffic modelling. *IEICE Transactions on Communications*, 88(4):1493–1502, 2005.

[81] Upendra Sharma, Prashant Shenoy, Sambit Sahu, and Anees Shaikh. A cost-aware elasticity provisioning system for the cloud. In *Proceedings of ICDCS*, pages 559–570. IEEE, 2011.

[82] David Starobinski and Moshe Sidi. Stochastically bounded burstiness for communication networks. *IEEE Trans. Information Theory*, 46(1):206–212, Jan. 2000.

[83] Maarten RC Van Oordt and Chen Zhou. The simple econometrics of tail dependence. *Economics Letters*, 116(3):371–373, 2012.

[84] Luis M Vaquero, Luis Rodero-Merino, and Rajkumar Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41(1):45–52, 2011.

[85] John Wilkes. More google cluster data. *Google research blog*, 2011.

[86] Ury Yechiali and Pinhas Naor. Queuing problems with heterogeneous arrivals and service. *Operations Research*, 19(3):722–734, 1971.

[87] Tadafumi Yoshihara, Shoji Kasahara, and Yutaka Takahashi. Practical time-scale fitting of self-similar traffic with Markov-modulated Poisson process. *Telecommunication Systems*, 17(1-2):185–211, 2001.

[88] Ming Yu and Mengchu Zhou. A model reduction method for traffic described by MMPP with unknown rate limit. *IEEE Communications Letters*, 10(4):302–304, 2006.

[89] Xinggong Zhang, Yang Xu, Hao Hu, Yong Liu, Zongming Guo, and Yao Wang. Profiling skype video calls: Rate control and video quality. In *INFOCOM, 2012 Proceedings IEEE*, pages 621–629. IEEE, 2012.