大规模网络服务系统行为异常的敏捷感知方法

章昭辉[1),2),3)] 崔 君[2)]

[1)](东华大学计算机科学与技术学院 上海 201620)

[2)](安徽师范大学数学计算机科学学院 安徽 芜湖 241003)

[3)](同济大学嵌入式系统与服务计算教育部重点实验室 上海 201804)

摘 要 大规模网络服务系统环境中,短时的大规模用户合法行为聚集会造成系统行为异常,使得系统可用性受到极大的损害.现有的系统异常检测方法大多适用于用户非法行为造成的系统异常.文中针对用户合法行为短时聚集引起的系统异常问题,提出一种大规模网络服务系统行为异常敏捷感知的方法.该方法包括系统异常敏捷感知模型和重复行为检测的 Petri 模型.基于"放大因子"的系统异常敏捷感知模型给出了系统异常的可预知性和异常系统行为的可定位性.即如果系统会在未来的某一时刻 t_2 发生异常,那么 t_1 时刻预期系统负载值要大于系统所能承受的最大负载值($t_2>t_1$).而且,该模型通过行为阻滞度可确定引起系统异常的系统行为.在敏捷感知模型的基础上,针对应用系统的异常检测,提出了基于带优先关系的颜色双变迁 Petri 网的重复行为检测模型及其系统异常敏捷感知算法.根据单位时间内用户行为数较小变化、缓慢增长和激增三种情况进行模拟实验,实验结果表明该方法可以有效地在系统异常发生之前提前感知,并能定位引起系统异常的系统行为.

关键词 网络服务系统;行为异常;敏捷感知;异常检测;Petri 网;云计算

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2017.00505

An Agile Perception Method for Behavior Abnormality in Large-Scale Network Service Systems

ZHANG Zhao-Hui[1),2),3)] CUI Jun[2)]

[1)](*School of Computer Science and Technology, Donghua University, Shanghai* 201620)

[2)](*School of Mathematics and Computer Science, Anhui Normal University, Wuhu, Anhui* 241003)

[3)](*The Key Laboratory of Embedded System and Service Computing of Ministry of Education, Tongji University, Shanghai* 201804)

Abstract In a large-scale network service system, the instantaneous gathering of a number of legal users can cause abnormality of system behaviors and vastly damage the system availability. Most of the existing methods can detect anomalies caused by illegal user behaviors. To detect those anomalies caused by legal users, this paper presents an agile perception method. This method includes an agile perception model of the system anomaly and a Petri Net model of repetitive behavior detection. The agile perception model based on Magnification Factor can propose the perception of system anomalies and the localizability of abnormal system behaviors. That is, the expectant system load with Magnification Factor at some time t_1 exceeds the maximum system load if the system is abnormal at some time t_2 ($t_2>t_1$) in the future. Furthermore, this model can also confirm the abnormal system behavior by the blocking degree of behaviors. Based on the agile perception model, this paper further presents a Petri Net detection model with priority relation and color double transition. To verify our method, we simulate a ticket booking system. According to the traffic graph of 12306 System, we simulate three change features of legal user

收稿日期:2016-01-08;在线出版日期:2016-06-19.本课题得到国家自然科学基金(61472004)、上海市"科技创新行动计划"高新技术领域项目(16511100903)、同济大学嵌入式系统与服务计算教育部重点实验室(2015)资助.**章昭辉**,男,1971 年生,博士,教授,硕士生导师,中国计算机学会(CCF)会员,主要研究领域为互联网信息服务、服务计算、云计算等.E-mail:zhzhang@163.com.**崔 君**,女,1992 年生,硕士研究生,主要研究方向为云计算、服务计算.

behaviors including increasing little, increasing slowly, and increasing sharply in unit time. The experimental results show that our method can effectively perceive a forthcoming system anomaly and locate it before its occurrence.

Keywords network service system; behavior abnormality; agile perception; abnormal detection; Petri net; cloud computing

1 引 言

随着通信技术、网络技术、分布式计算技术等日新月异地发展,基于网络的服务系统为满足用户需求带来了便捷.但是,系统本身的大规模性、全方位性、复杂性等特征越来越突出;同时,系统的用户群体也随之呈现出规模巨大、用户需求多样化的特征.大规模网络系统的服务能力和服务质量与大规模用户群体的需求满足之间的矛盾也越来越凸显.尤其是当某些扰动事件发生时,系统会吸引大规模的用户群体以及异常行为用户(如使用抢票软件、"黄牛"囤票软件,其本质上也是聚集大规模用户的一种形式,只是其购票流程实现自动化)参与系统,用户群体的刚性需求引起对资源的过度竞争和非法竞争,从而使得系统的可用性急剧下降,系统资源和用户资源(时间等)极大浪费.

基于网络的服务系统为了满足用户需求,新一代软件范型不断涌现,如普适计算软件、超大规模软件系统、网构软件等[1].与传统计算机软件系统相比,互联网服务系统所表现出的大规模性、开放性、复杂性等特征,使得软件自适应对系统感知提出更高的要求[2-3].网络系统环境的开放性可能使得大规模网络服务软件系统在开发设计时就存在固有的缺陷,如不可预见的用户行为增长与有限的系统负载之间的不平衡.这些潜在缺陷会在软件系统运行时随着用户的增加显现出来,对系统的可用性造成极大的损害.12306 购票系统曾经在春节前瘫痪就是一个典型例子.系统一开始是满足用户需求的,但是随着用户量和用户向系统重复提交请求行为数的激增,系统负载随之激增,导致系统性能急剧下降、甚至崩溃.短时用户量激增以及用户施加于系统的重复行为数激增是大规模网络服务系统的可用性造成严重伤害的重要成因.因此,要保证系统能够在动态开放环境下持续、高质量地提供服务,系统需要在运行前期或是运行中提前快速感知自身异常并提供相应地异常预防措施.然而,现有的系统状态感知或检测研究[4-5]大多考虑的是基于用户非法行为产生的系统异常,并未考虑用户合法行为短时聚集对系统所造成的异常;或是从系统流量、事件日志表征现象中挖掘出异常,不能及时有效地在异常发生之前感知.

用户合法行为对系统造成异常的敏捷感知与检测的关键在于,如何在系统异常发生之前就能快速感知系统异常并定位异常的系统行为.本文主要有以下贡献:

(1)针对用户量和用户向系统重复提交请求行为数的短时激增对系统的影响,建立了系统异常的敏捷感知模型.该模型提出了"放大因子"的概念,通过建立"放大因子"与系统预期负载的关系模型将用户量与用户重复提交的行为数对系统负载的影响放大来考察系统负载的变化趋势.在 t 时刻,用户提交的正常行为数在"放大因子"的作用下所需的预期系统负载值若超过系统所能承受的最大负载值,说明在未来的某一时刻系统会出现异常,且异常的地方在于该时刻所有系统行为中阻滞度比重最大的系统行为.

(2)在敏捷感知模型的基础上,建立了系统重复行为检测方法.通过对系统行为进行 Petri 网建模,分析其系统 Petri 网结构,确定关键节点.通过对关键节点设置检测机制,即利用 PCDPN(带优先关系的颜色双变迁 Petri 网)对系统模型进行行为重构,并将感知模型运用到所建立的 PCDPN 网系统行为模型中,提出相应的检测算法,通过实例及实验验证该方法的有效性.

本文第 2 节综述系统异常感知与检测的国内外相关研究工作;第 3 节阐述系统异常敏捷感知模型,并给出该模型对异常行为的可预知性和可定位性结论;第 4 节给出对实际系统中重复行为的 Petri 网检测模型及算法;第 5 节给出模型实例及有效性的实验验证;第 6 节给出本文工作与其他工作比较的相关讨论;第 7 节给出本文的工作总结和下一步工作.

2 相关工作

为了提高软件的可用性以及系统服务的质量,

对软件自适应的研究现已成为关注热点. 软件自适应包括感知、决策、执行 3 个环节[1]. 目前国内外对感知环节的研究主要从两个方面展开：一是基于环境上下文处理，获取和处理软件的环境信息[6]. 软件系统的开放性和动态性导致了软件环境不断的变化，因此对环境上下文的研究也开始更多地强调环境信息的显式化. 丁博等人[7]概述了普适计算中间件技术对上下文信息的获取方式. 朱可可[8]设计了面向环境感知的自适应中间件模型，自适应服务层通过从传感器感知外部环境从而进行自适应选择反馈到应用层. 张慧[9]提出了面向语用网的上下文感知系统，通过设计一个面向语用网的上下文感知系统框架，使之可以动态地知识表达与推理，能够更加准确地理解用户的真实意图. Chen 等人[10]将上下文定义为环境状态和场景集合，它们分为可能影响应用行为的上下文、由应用展现且用户对之感兴趣的上下文两类. Yang 等人[11]提出利用“建模-规约-检测”的分布异步环境感知框架来检测异步动态属性的普适计算环境. Cheng[12]提出了 Rainbow 模型，进一步将软件自适应拓展到分布环境下，通过在结点之间建立严格的层次控制关系来实现群体自适应. 二是基于软件的监测系统，获取软件的内部状态信息. Robinson[13]通过用户的需求分析和软件在运行时的监测技术相结合，实现对 Web 服务行为的监控，保证电子商务的可用性. Garlan 等人[14]使用探针和量规三层监控机制来实现对软件体系结构的监测，监测系统体系结构的变化. Mi 和 Wang 等人[15]针对用户请求的结构提出了一种可以快速诊断导致系统性能下降的原因的检测方法.

从感知异常行为的角度来看，大多数的检测技术是通过建立系统、用户或程序的正常行为轮廓，比较和匹配被监测对象的实际行为与正常行为轮廓来实现的[16]. 谢逸等人[17]提出一种基于 Web 用户访问行为的异常检测方案，用于检测应用层上的分布式拒绝服务攻击，并以具有非稳态流特性的大型活动网站为例，进行应用研究. 冯震[18]提出基于建立正常用户行为流量特征的方法，与之比对来检测可能对网络本身或其承载的数据业务造成危害的异常行为，并根据异常行为的特征来采取相应措施，降低这些异常行为所造成的危害，保证网络的安全运行. Huang 等人[19]提出一种级联模式用于分布式网络的在线监测异常事件. Li 等人[20]提出异常相关性分析方法，从全网的流量分析异常数据，揭示攻击行为变化的相关性.

从软件系统的故障检测的角度来看，在大规模网络环境下对分布式软件进行异常检测和故障诊断显得十分困难[21]. 近年来故障检测的研究主要从结点或是结构等方面开展. Smith 等人[22]利用贝叶斯方法对结点数据进行降维并通过 PCA 方法解析出影响结点健康状况的主要指标，最后根据结点之间的差异来检测故障结点. Bertier 等人[23]提出了一种的分层检测结构，通过将节点分成若干组，在每个组内设置一个 Leader 节点，负责完成对组内节点的故障检测，并且可以根据应用程序需求通过共享服务自适应调整服务质量. Shi 等人[24]也提出了一种类似的层次结构故障检测协议，包含了不可靠的故障检测技术服务和微处理器体系结构的一些想法，并重点讨论了这种结构的良好的可伸缩性和灵活性. Wang 等人[25]在云计算的环境中提出了一种基于熵的异常测试方法模型. Basile 等人[26]提出带标记的时间 Petri 网来动态的评估系统的状态和故障检测.

综上所述，现有的系统异常感知和检测主要考虑的是用户非法行为对系统所造成的异常，而未考虑用户合法行为对系统的影响. 通过表征检测异常的方法缺乏考虑实时预知用户合法行为对系统的可用性造成的损害. 这种影响主要是开放环境下的网络服务系统对用户行为的不可预见性和不可控性所造成的. 而系统行为对系统本身来说应当是可控性的. 但这种可控性必须是系统尽早感知潜在异常的发生，即系统异常的可控性是以系统对异常的可预见性为前提. 本文提出的大规模网络服务系统的异常行为敏捷感知方法，是针对用户合法行为短时聚集对系统行为所造成的异常，通过系统行为流程中微小的负载变化，在系统运行前期或系统运行中提前快速地感知系统异常行为，以达到利用系统行为的可预见性解决用户行为的不可预见性，从而为系统服务调整提供支持.

3　系统异常敏捷感知模型

3.1　系统异常

在大多数系统运行正常情况下，系统是可以满足用户需求的. 但当某一特定时间或是受事件、其他用户行为、环境等因素影响时，大规模网络服务系统性能及系统行为与用户需求及用户行为容易发生不相适配的状况，即系统异常. 例如，火车票网上购票

系统每到节假日,尤其是春节,用户群体规模急剧扩大,系统就呈现出不堪使用的状态甚至瘫痪.

从用户使用系统的流程看,平时用户完成一次购票过程很顺利.但是当系统并发用户量激增时,系统资源会被占用,导致系统的流畅度下降,有些用户会因自身原因在此次流程中的某些环节不断重复操作(如不断刷新页面、重复提交等),这使得系统负担加重,系统的可用性极大下降,造成系统异常.例如,每年双十一活动时,电商平台聚集了大量的用户,有数据表明双十一这一天用户购买物品的一系列过程会比平常慢很多,尤其是在付款环节,用户付不了款或是要不断重复付款操作,用户对购物过程的体验度也比平时差.

由此可知,在系统运行中,系统负载增加主要有两个原因,一是用户量的增加,另一个是用户提交的请求行为数增加.而用户量和用户提交的请求行为数的增加则是因为系统群体用户行为中有行为噪音的存在.这种行为噪音在大规模的网络环境中具体表现为个体用户的行为偏差会影响其他用户的行为序列,并且这种影响会扩大到群体用户,造成群体用户的行为偏差,最终导致系统异常.本文主要考虑用户重复提交请求的行为偏差,并将这种具有不可控性的行为偏差定义为放大因子,以此建立系统异常敏捷感知模型.

3.2 系统异常敏捷感知模型

定义 1. t 时刻系统中用户正常提交的总的行为数等于用户量,即 $Bf_t=S_t$.其中 S_t 表示 t 时刻的大规模网络服务系统的用户数.

定义 2. t 时刻系统中用户提交的总的行为数 B_t,是 t 时刻系统中所有用户正常提交的行为数和重复提交的行为数的总和,即 $B_t=Bf_t+Br_t$.其中 Bf_t 表示 t 时刻系统中用户正常提交的总的行为数;Br_t 表示 t 时刻系统中用户重复提交的总的行为数.

定义 3. t 时刻系统的实际负载 C_t,是用户在当前时刻 t 提交的总的行为数对应的一个系统负载,即 $C_t=B_t\times a$.其中 $a(a\geq 1)$ 表示一个用户提交的一个请求行为所需要的系统行为的负载.

定义 4. 系统重复行为阻滞系数 β_1,是 t 时刻所有用户重复提交的行为数 Br_t 占总的系统行为数 B_t 的比重,即 $\beta_1=Br_t/B_t$.

定义 5. 系统行为阻滞系数 β_2,是 t 时刻所有用户正常提交的行为数 Bf_t 占总的系统行为数 B_t 的比重,即 $\beta_2=Bf_t/B_t$.

定义 6. 放大因子(NTR),是单位时间内用户量和用户重复提交系统行为数的变化量,即 $NTR_t=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t$.其中 ΔBr_t 表示在趋近 t 时刻的用户重复提交的总的行为数的变化量;ΔS_t 表示在趋近 t 时刻的用户数的变化量.

定义 7. t 时刻预期的系统负载 Cb_t,是用户在当前时刻 t 正常提交的总行为数在放大因子的作用下所需要的系统负载,即 $Cb_t=NTR\times Bf_t\times a$.

定义 8. 系统行为阻滞度 w_{it}.$\{b_1,b_2,b_3,\cdots\}$ 表示系统的行为序列,t 时刻系统行为 b_i 总数占总的系统行为数 B_t 的比重,即 $w_{it}=\sum b_i/B_t$.如果 t 时刻系统行为 b_i 的行为阻滞度最大,说明行为 b_i 在系统中是不稳定的.

假设 1. 大规模网络服务系统在设计之初都有一个自身可以承受最大的系统负载 $C_{\max}$.

假设 2. 在 t_2 时刻,系统因 $\Delta S_t/\Delta t$ 或 $\Delta Br_t/\Delta t$ 的激增而产生异常,即当 $C_{t_2}>C_{\max}$ 时,$\exists t$,$t<t_2$,有 $\lim\limits_{\Delta t\to 0}\Delta S_t/\Delta t=\infty$ 或 $\lim\limits_{\Delta t\to 0}\Delta Br_t/\Delta t=\infty$.

定理 1. 可预知性.在假设 1、2 的条件下,如果系统在 t_2 时刻系统会发生异常,那么根据定义 1~7 的模型,在 t_1 时刻可预测系统会发生异常,即 t_1 时刻系统的预期负载超过系统的最大负载,其中 $t_1<t_2$.

证明. 假设在任意 $t(t<t_2)$ 时刻不能预测到系统会发生异常,即 $Cb_t\leq C_{\max}$.其中 Cb_t 是系统 t 时刻的预期负载,$C_{\max}$ 是系统的最大负载.

根据定义 1~7,则有

$NTR\times Bf_t\times a\leq C_{\max}<C_{t_2}$ 成立.即

$NTR\times Bf_t\times a<C_{t_2}=B_{t_2}\times a$.

可知 $\dfrac{Br_t}{B_t}\times\dfrac{\Delta Br_t}{\Delta t}+\dfrac{Bf_t}{B_t}\times\dfrac{\Delta S_t}{\Delta t}<\dfrac{B_{t_2}}{Bf_t}$,

设 $y=\min\{Br_t,Bf_t\}$,则有

$$\frac{y}{B_t}\times\left(\frac{\Delta Br_t}{\Delta t}+\frac{\Delta S_t}{\Delta t}\right)<\frac{B_{t_2}}{Bf_t},$$

即 $\dfrac{\Delta Br_t+\Delta S_t}{\Delta t}<\dfrac{B_tB_{t_2}}{y\times Bf_t}<\dfrac{B_{t_2}^2}{y^2}$ 成立.该不等式最右边是一个定数.当 $\Delta t\to 0$ 时,这与假设 2 矛盾.

因此,$\exists t_1$,$t_1<t_2$,使得 $Cb_{t_1}>C_{\max}$.即系统如果发生异常则可提前感知. 证毕.

定理 2. 可定位性.如果系统异常是因 $\Delta St/\Delta t$ 或是 $\Delta Br_t/\Delta t$ 的激增引起的,那么可定位到某个系统异常行为.

证明. 根据定理 1 可知在 t_1 时刻系统的预期负载大于系统最大负载.根据定义 3,此时用户行为对应于相应的系统行为 b_i.由于该预期负载根据 t_1 时刻的用户行为数 $C_t=B_t\times a$ 得到.根据定义 8,可计算出阻滞度最大的系统行为 b_i. 证毕.

3.3　案例分析

以某网购大规模网络服务系统为例来说明系统异常感知敏捷模型的原理，如图 1 所示是某网购系统流程.

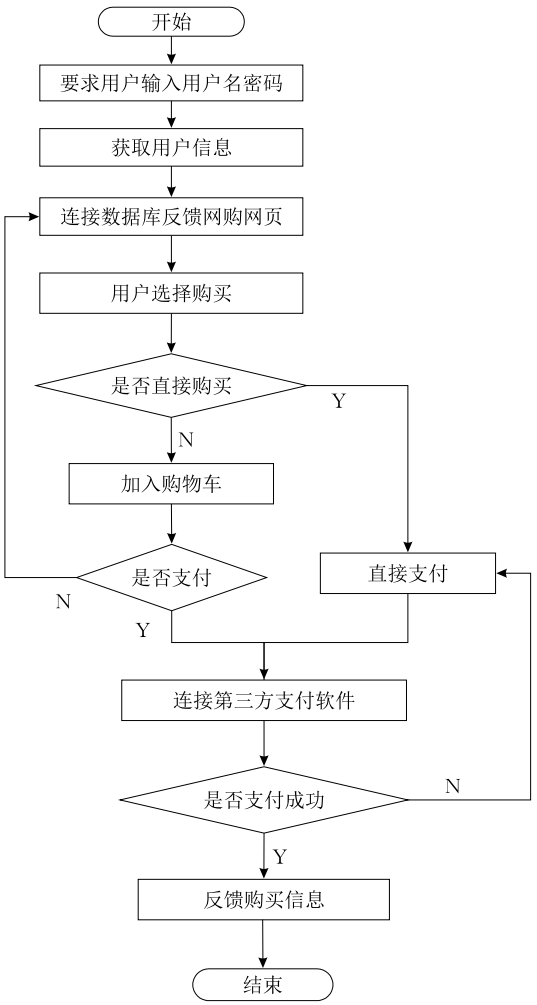

图 1　某网购系统流程

假设对网购系统流程中的用户登录行为、浏览网页行为、加入购物车行为、支付行为进行关键监测，检测其上述 4 种关键系统行为总的用户提交正常行为数和重复行为数. 具体数据假设如表 1 所示.

表 1　检测系统总行为数数据表

	Time	Bf_t	Br_t	B_t
t_1	9:00	12083	7953	20036
t_2	9:10	15937	12394	28331
t_3	9:20	22671	17568	40239
t_4	9:30	33755	21052	54807
t_5	9:40	38272	21953	60225

假设该网购系统所能承受的最大系统负载 $C_{\max}$ 为 60000，一个用户提交的一个请求行为所需要的系统负载 a 为 1；从表 1 数据得出：

当 $t_4=9:30$，

$C_{t_4}=B_{t_4}\times a=54807\times 1$

$=54807<C_{\max}=60000$，

$\beta_1=Br_{t_4}/B_{t_4}=21052/54807\approx 0.38$，

$\beta_2=Bf_{t_4}/B_{t_4}=33755/54807\approx 0.62$，

$NTR_{t_4}=\beta_1\Delta Br_{t_4}/\Delta t+\beta_2\Delta S_{t_4}/\Delta t$

$=0.38\times 3484/3600+0.62\times 11084/3600$

≈ 2.28，

$Cb_{t_4}=NTR\times Bf_{t_4}\times a$

$\approx 2.28\times 33755\times 1$

$\approx 76961>C_{\max}=60000$；

当 $t_5=9:40$，

$C_{t_5}=B_{t_5}\times a=60225\times 1$

$=60225>C_{\max}=60000$；

即当 $Time=9:30$ 时，$Cb_{t_4}>C_{\max}>C_{t_4}$，系统发出预警，且 $t_4<t_5$.

表 2　9:30 检测各行为数据表

行为	含义	*Time*	Bf_t	Br_t	B_t
b_1	用户登录行为	9:30	7124	3232	10356
b_2	反馈网页行为	9:30	6987	3167	10154
b_3	加入购物车行为	9:30	7646	4551	12197
b_4	支付行为	9:30	12998	9105	22103
b_5	系统总的行为数	9:30	33755	21052	54807

从表 2 数据得出：当 $Time=9:30$ 时，$w_{4t}=\sum b_4/B_t=22103/54807\approx 0.40$ 最大，所以系统异常行为是 b_4，即支付行为.

4　重复行为检测的 Petri 网模型及算法

本文第 3 节所提出的系统异常敏捷感知模型对快速感知用户合法行为短时聚集引起系统异常问题提供了理论支持，本节将阐述如何将敏捷感知模型应用到实际系统的异常行为检测中. 在实际的应用系统中，敏捷感知模型应用实现的关键是如何检测系统重复行为. 检测系统重复行为首先要对系统进行建模，再对系统的关键节点设置检测机制. 系统建模工具有形式语言与自动机、Petri 网等，同其他系统模型相比较，对系统并发的贴切描述是 Petri 网的独特优势. 下面讨论通过对系统进行 Petri 网建模，建立可检测重复行为的 Petri 网系统模型及其算法，并将敏捷感知模型应用于该检测算法中.

4.1　带优先关系的颜色双变迁 Petri 网

传统 Petri 网是由库所、变迁和流关系 3 个基本

结构构成，是描述离散事件动态系统的有力工具.下面先简述 Petri 网的基本概念，然后给出带优化关系的颜色双变迁 Petri 网模型.

定义 9. 网[27]. 三元组 $N=(P,T,F)$称为一个网，当且仅当

(1) P 是一个有限库所集，T 是一个有限变迁集；

(2) $P\bigcup T\neq\varnothing$，$P\bigcap T=\varnothing$；

(3) $F\subseteq(P\times T)\bigcup(T\times P)$是一个弧集.

定义 10. Petri 网[27]. $PN=(N,M)$称为一个 Petri 网，当且仅当

(1) $N=(P,T,F)$是一个网；

(2) $M:P\rightarrow\{0,1,2,\cdots\}$是 PN 的一个标识函数；

(3) 引发规则：

① $\forall p\in{}^{\cdot}t:M(p)>0$，则称变迁 t 在标识 M 使能，记为 $M[t\rangle$；

② 若变迁 t 在标识 M 使能，则 t 可以引发且引发后产生一个后继标识 M'，记作 $M[t\rangle M'$，其中

$$M'(p)=\begin{cases}M(p)+1, & p\in t^{\cdot}-{}^{\cdot}t\\ M(p)-1, & p\in{}^{\cdot}t-t^{\cdot}\\ M(p), & \text{其他}\end{cases}.$$

定义 11. 冲突关系[27]. 设$(P,T;F,M_0)$为一个基本网系统，$t_1,t_2\in E$，M 是$\sum$的一个情态. 如果满足：

(1) $M[t_1\rangle$且 $M[t_2\rangle$；

(2) $M[t_1\rangle M'\rightarrow\neg M'[t_2\rangle\wedge M[t_2\rangle M''\rightarrow\neg M''[t_1\rangle$

则称事件 t_1 和 t_2 在情态 M 处于冲突关系.

颜色双变迁 Petri 网是传统 Petri 网的基础上扩展了颜色以及变迁得到的高级 Petri 网. Gao 等人[28]提出的颜色双变迁的 Petri 网，通过颜色集、变迁集以及弧表达式等增强了控制逻辑表达能力. 针对变迁的不同优先级的问题，Best 等人[29]提出了带优先关系的 Petri 网. 本文在此基础上，为解决系统重复行为的检测，提出一种带优先关系的颜色双变迁 Petri 网.

定义 12. 带优先关系的颜色双变迁 Petri 网是一个 $PCDPN=(\Sigma,P,T,Q;F,C,\rho)$；其中

(1) Σ 是有限颜色集合，服从泊松分布函数，$\Sigma\sim P(\lambda)$；

(2) P 是库所集，$P=\{p_1,p_2,\cdots,p_n\}(n\geq0)$；

(3) T 是服务行为变迁集，$T=\{t_1,t_2,\cdots,t_k,t_{k+1},\cdots,t_{k+i}\}(k\geq0,i\geq0)$，其中 t_1 到 t_k 是系统流程本身固有的服务行为变迁，t_{k+1} 到 t_{k+i} 是用户重复提交系统行为时所需触发的服务行为变迁；

(4) Q 是控制行为变迁集，$Q=\{q_1,q_2,\cdots,q_m\}(m\geq0)$；

(5) F 是有向弧集，$F=F_C\bigcup F_D$，$F_C\subseteq(P\times T)\bigcup(T\times P)$，$F_D\subseteq(P\times Q)\bigcup(Q\times P)$；

(6) C 是颜色函数，$C:P\rightarrow Z$(非负整数集)，库所 P 中的托肯值属于有限颜色集合 Σ，且用户在时间 t 内到达系统的数量可以服从泊松分布[30]；

(7) 优先关系 ρ 表示一个变迁的偏序关系，即 $(q_i,t_i)\in\rho$，定义控制行为变迁 q_i 的优先级高于服务行为变迁 t_i 的优先级.

4.2 PCDPN 模型

4.2.1 Petri 网的 4 种基本结构

基于 Petri 网的系统模型主要由顺序、并发、选择、循环这 4 种基本结构构成[31]，如图 2 所示.

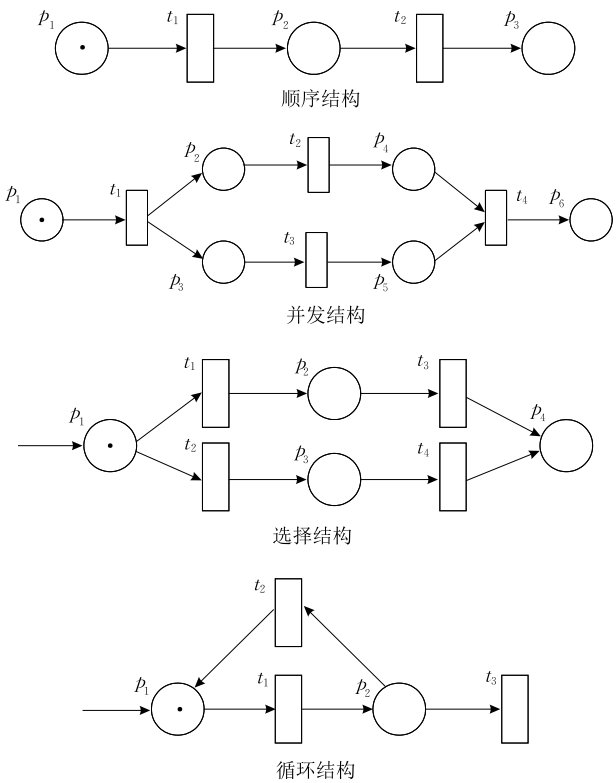

图 2 Petri 网的 4 种基本结构

4.2.2 PCDPN 的 4 种基本结构

下面利用 PCDPN 分别对大规模网络系统模型的几种基本结构进行建模，如图 3 所示.

图 3 中有两种变迁，一种是系统服务行为流程变迁 t，一种是控制行为变迁 q. 当用户提交重复系统行为时，触发代表重复行为的控制行为变迁 q，此时正常行为流程的服务行为变迁 t 和控制行为变迁 q 属于冲突关系，优先触发控制行为变迁 q，使用户回到之前的行为流程中，并且重复行为数加 1，系统

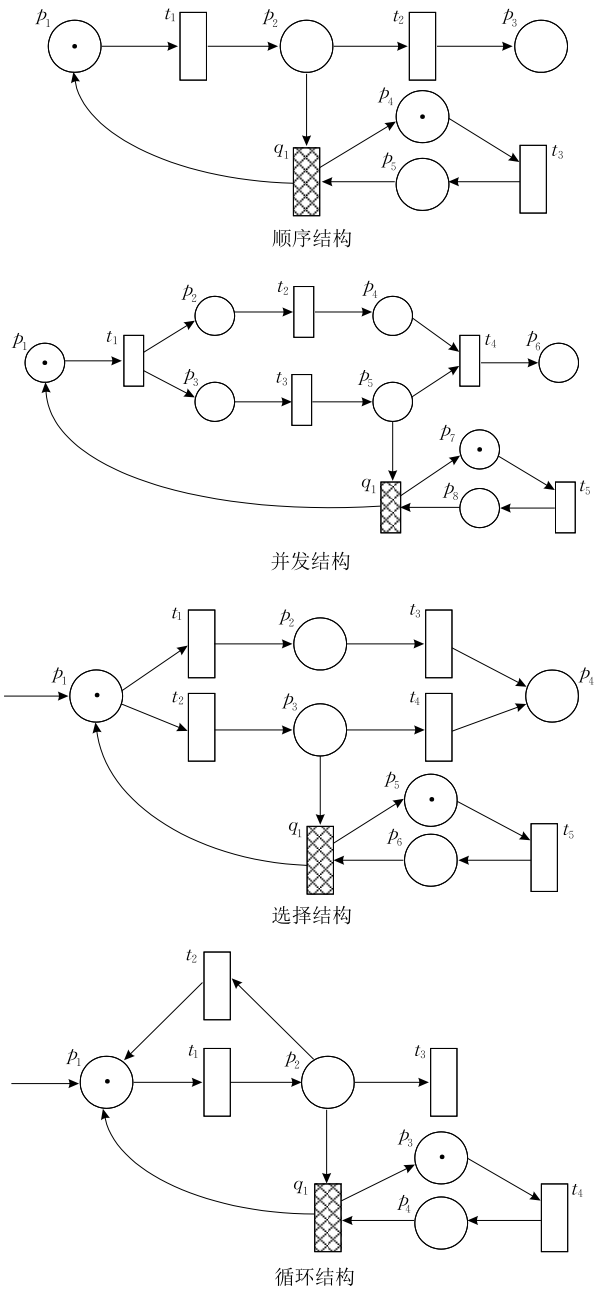

图 3　PCDPN 模型的 4 种基本结构

行为流程库所中的托肯值减 1.

4.3　重复行为检测的 PCDPN 模型

4.3.1　顺序结构

对系统行为流程进行 Petri 网建模，如果有关键行为节点属于顺序结构，则对该关键节点进行重构，转变成 PCDPN 模型，以此对该节点进行重复行为检测.

以图 3 顺序结构为例，$p_1 \sim p_5$ 是库所集，表示状态. 当用户重复提交一个行为，服务行为变迁 t_3 触发. 此时服务行为变迁 t_2 和控制行为变迁 q_1 在此情态下同时满足触发条件，属于冲突关系，因控制行为变迁 q_1 的优先级大于服务行为变迁 t_2，控制行为变迁 q_1 优先触发，且重复行为数加 1，系统行为流程库所中的托肯值减 1.

4.3.2　并发结构

对系统行为流程进行 Petri 网建模，如果有关键行为节点属于并发结构，则对该关键节点进行重构，转变成 PCDPN 模型，以此对该节点进行重复行为检测.

以图 3 并发结构为例，$p_1 \sim p_6$ 是库所集，表示状态. 当用户重复提交一个行为，服务行为变迁 t_5 触发. 此时服务行为变迁 t_4 和控制行为变迁 q_1 在此情态下同时满足触发条件，属于冲突关系，因控制行为变迁 q_1 的优先级大于服务行为变迁 t_4，控制行为变迁 q_1 优先触发，且重复行为数加 1，系统行为流程库所中的托肯值减 1.

4.3.3　选择结构

对系统行为流程进行 Petri 网建模，如果有关键行为节点属于选择结构，则对该关键节点进行重构，转变成 PCDPN 模型，以此对该节点进行重复行为检测.

以图 3 选择结构为例，$p_1 \sim p_6$ 是库所集，表示状态. 当用户重复提交一个行为，服务行为变迁 t_5 触发. 此时服务行为变迁 t_4 和控制行为变迁 q_1 在此情态下同时满足触发条件，属于冲突关系，因控制行为变迁 q_1 的优先级大于服务行为变迁 t_4，控制行为变迁 q_1 优先触发，且重复行为数加 1，系统行为流程库所中的托肯值减 1.

4.3.4　循环结构

对系统行为流程进行 Petri 网建模，如果有关键行为节点属于循环结构，则对该关键节点进行重构，转变成 PCDPN 模型，以此对该节点进行重复行为检测.

以图 3 循环结构为例，$p_1 \sim p_4$ 是库所集，表示状态. 当用户重复提交一个行为，服务行为变迁 t_4 触发. 此时服务行为变迁 t_3 和控制行为变迁 q_1 在此情态下同时满足触发条件，属于冲突关系，因控制行为变迁 q_1 的优先级大于服务行为变迁 t_3，控制行为变迁 q_1 优先触发，且重复行为数加 1，系统行为流程库所中的托肯值减 1.

4.4　算法 PCDPN

根据 PCDPN 模型，将所有关键行为节点的所检测出来的重复行为数进行累加得到系统总的重复行为数. 其中用户正常提交的行为数为系统行为流程库所的托肯值. 然后根据定义 1～8 的敏捷感知模型，计算系统的预期负载和行为的阻滞度. 相应的系

统感知检测算法 PCDPN 如下所示.

算法 PCDPN. 系统异常行为敏捷感知检测.

输入:S_t,B_t,Br_t,Bf_t,a,n; //在 t 时刻系统监测或计算出用户量 S_t 以及所有用户在当前时刻 t 提交的总的行为数 B_t 和重复提交的行为数 Br_t 及用户行为负载 a

输出:系统异常预警和异常行为

1. WHILE $t>0$;
2. $C_t=B_t\times a$;

//计算所有用户在当前时刻 t 提交的总的行为数对应的一个实际的系统负载 C

3. $\beta_1=Br_t/B_t$; $\beta_2=Bf_t/B_t$;
   $NTR_t=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t$;

//计算当前 t 时刻的放大因子的值

4. $Cb_t=NTR\times Bf_t\times a$;

//计算所有用户在当前 t 时刻提交的正常行为数在放大因子的作用下所需要的预期系统负载

5. IF $Cb_t>C_{\max}$ THEN
6. 预警系统在未来的某个时刻会发生异常;

//定位可能发生异常的系统行为

7. $i\leftarrow 1$;
8. WHILE $i<n$ //计算 t 时刻所有行为 b_i 的阻滞度
9. $w_{it}=\sum b_i/B_t$; $i=i+1$;
10. END WHILE
11. $w_{\max}\leftarrow w_{1t}$; $k\leftarrow 0$; $i\leftarrow 2$;
12. WHILE $i<n$
13. IF $w_{it}>w_{\max}$ THEN
14. $w_{\max}\leftarrow w_{it}$
15. $k\leftarrow i$
16. END IF
17. $i=i+1$
18. END WHILE //找出阻滞度最大 w_{kt}
19. PRINT 系统在未来的某个时刻会发生异常,且导致异常的行为是 b_k;
20. END IF
21. $t=t+m$ //检测时间间隔为 m
22. END WHILE
23. 算法结束.

5 实例与实验

5.1 购票系统的 PCDPN 模型

某些购票系统就是典型的大规模网络服务软件系统,例如 12306 购票系统. 每逢春运,12306 购票系统的用户量就会突然增多,以及抢票软件的使用,使得系统负载的增大,导致系统变慢甚至崩溃. 模拟某购票系统的基本流程(如图 4 所示),在其购票系统 Petri 网模型的基础上构建重复行为检测的 PCDPN 网模型,并将系统异常行为敏捷感知检测算法运用到模型中,通过实验验证重复行为检测模型的有效性,且可以敏捷感知到系统在未来的某个时间是否会发生异常.

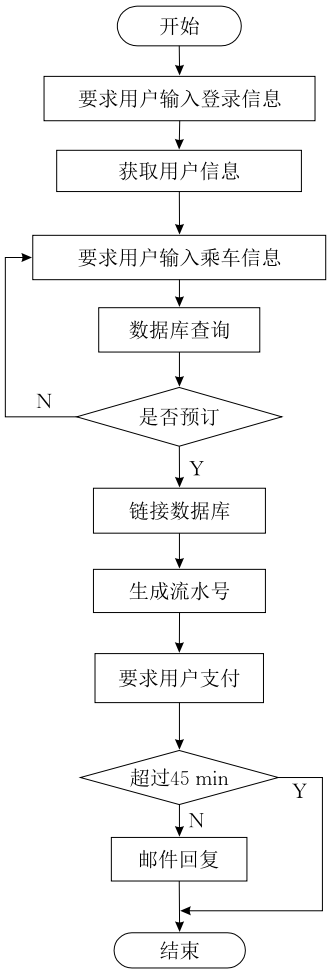

图 4 某购票系统的基本流程

根据购票系统的流程图进行 Petri 建模,如图 5 所示.

对于图 5 中各变迁的含义如表 3 所示.

表 3 图 5 模型中变迁所对应的行为活动

变迁	行为活动
t_1	要求用户输入登录信息
t_2	获取用户信息
t_3	要求用户输入乘车信息
t_4	系统数据库查询
t_5	用户是否预订
t_6	用户没有预订重新返回查询页面
t_7	连接数据库
t_8	订单生成流水号
t_9	要求用户支付
t_{10}	判断用户支付是否超过 45 min
t_{11}	系统自动邮件回复用户订单详情
t_{12}	用户订票结束

对图 5 的购票系统的 Petri 网模型进行分析，确定登录行为、查询行为、预订行为和支付行为是其关键行为，利用 PCDPN 对这几个关键行为进行行为流程建模，如图 6 所示.

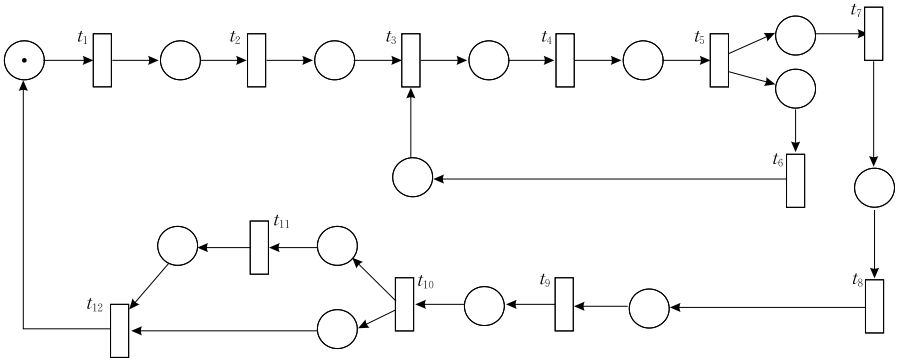

图 5　购票系统的 Petri 网模型

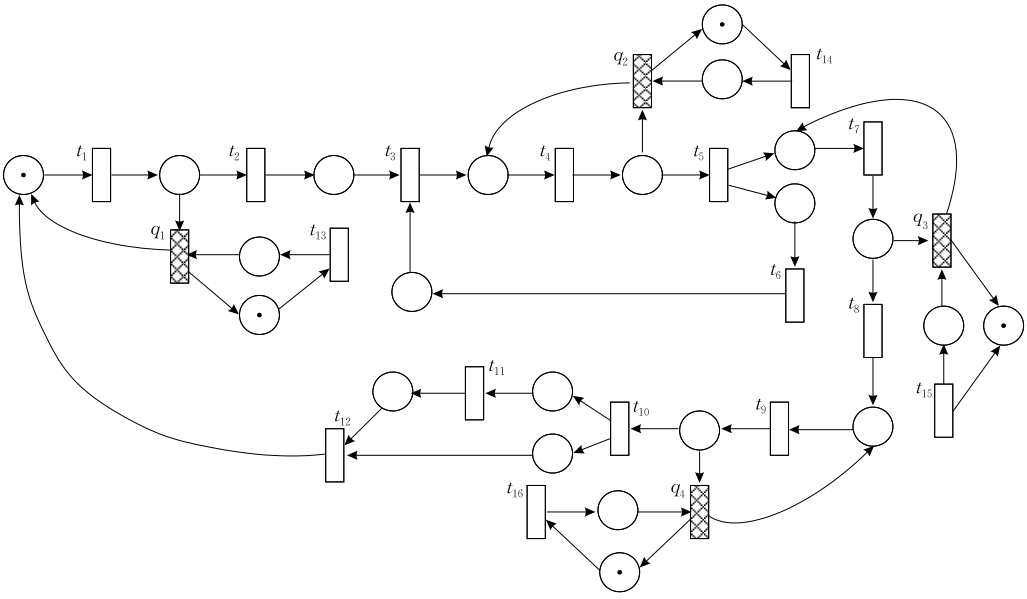

图 6　购票系统的 PCDPN 模型

对于图 6 中 $t_1 \sim t_{12}$ 变迁的含义见表 3，$t_{13} \sim t_{14}$，$q_1 \sim q_4$ 变迁的含义如表 4 所示.

表 4　图 6 模型中 $t_{13} \sim t_{14}$，$q_1 \sim q_4$ 变迁所对应的含义

变迁	含义
$t_{13} \sim t_{14}$	重复行为的服务变迁
$q_1 \sim q_4$	控制行为变迁

5.2　实验设计

本文通过对购票系统的 PCDPN 模型所示的流程进行模拟，其中对登录、查询、预订和支付这几个模块进行重点考察，并对每个模块设置了监控，主要包括图 6 中的控制行为变迁 $q_1 \sim q_4$，采集行为数等相关数据，其中系统服务行为流程中的库所的托肯值表示用户量，用户重复行为数由累加得到.

模拟系统的行为序列如图 7 所示. 图 7 中主要行为及数据的模拟如下：

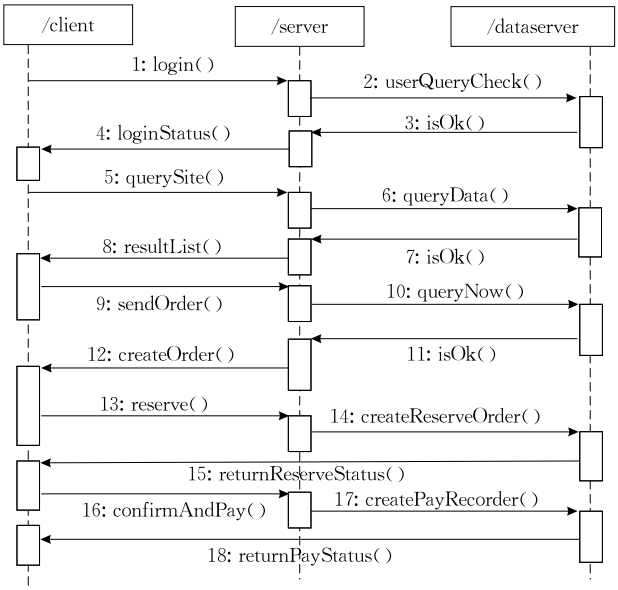

图 7　模拟系统的行为序列

(1)用户登录,此环节需要完成用户名、密码、IP 的模拟;

(2)服务端响应登录请求,接收模拟的用户名、密码、IP,进行模拟校验;

(3)登录完成后,查询,此环节需要完成起止地点、时间的模拟;

(4)响应查询请求,接收模拟的查询的起止地点、时间,模拟数据库查询,返回符合条件列表;

(5)模拟预订请求,此环节需要确定完成订单提交的模拟;

(6)响应预订请求,根据查询条件返回此时的提交的订单剩余量,提交进行排队,计算剩余时间,响应预定请求,返回状态码;

(7)模拟支付请求,此环节需要车次、支付节点的模拟;

(8)响应支付请求,连接第三方服务代理,返回支付状态.

由于无法获取到购票系统的实际数据,本文设置一个数据发生器,使其对各个模块不断施加用户提交的正常行为数和重复行为数,造成模拟系统负载的增加.计算模拟数据,为保证数据的有效性,与 12306 系统某一时段的具体流量图作对比(如图 8 所示①),使之符合 12306 购票网站的流量特征,即短时聚集性.

对 12306 购票网站的流量进行分析,发现其在变化的过程中有着单位时间内,系统中总行为数变化不大、总行为数变化缓慢以及总行为数激增的 3 种不同的流量特征.根据这 3 种 12306 购票网站的流量特征,分别模拟 3 组不同的用户及用户行为数据进行实验.定义负载在[0,60 000)区间内属于良好服务区,[60 000,120 000)区间内属于可用性受损区,[120 000,+∞)属于服务不可用区.将这 3 组实验数据代入系统异常行为敏捷感知检测算法,通过 JavaScript 和 Java 编写模拟系统,利用画图插件 Highcharts 表现 3 个实验效果图.

图 8 火车票定票网站流量图

5.3 实验结果分析

第 1 组实验数据见表 5,把表 5 的数据运用到感知检测算法中,得到效果如图 9 所示.从图 9 可以看出,实际负载随着时间的推移变化比较平缓,相比之下系统预期负载的变化趋势比较明显,且系统预期负载变化量比较大的时刻(结合表 5 的数据)和系统总的行为数变化量较大的时刻是重合的.

表 5 第 1 组实验数据

Time	B_t	Bf_t/S_t	Br_t	*Time*	B_t	Bf_t/S_t	Br_t	*Time*	B_t	Bf_t/S_t	Br_t
13:00	91 890	52 758	39 132	15:00	79 435	46 526	32 909	17:00	23 256	17 892	5364
13:10	88 068	49 548	38 520	15:10	64 561	39 525	25 036	17:10	25 308	19 044	6264
13:20	90 636	47 790	42 846	15:20	60 954	39 236	21 718	17:20	25 074	18 888	6186
13:30	89 358	48 534	40 824	15:30	55 943	36 524	19 419	17:30	25 278	18 492	6786
13:40	91 314	46 938	44 376	15:40	48 672	34 992	13 680	17:40	25 122	18 108	7014
13:50	89 922	48 324	41 598	15:50	42 018	30 114	11 904	17:50	25 254	18 984	6270
14:00	97 274	48 497	48 777	16:00	31 350	23 892	7458	18:00	25 092	18 312	6780
14:10	90 288	52 482	37 806	16:10	32 442	23 280	9162	18:10	25 314	18 456	6858
14:20	94 734	50 112	44 622	16:20	31 260	21 810	9450	18:20	25 062	18 264	6798
14:30	90 990	52 188	38 802	16:30	32 076	21 564	10 512	18:30	24 570	18 576	5916
14:40	94 356	52 536	41 820	16:40	31 770	20 922	10 848				
14:50	86 543	55 258	31 285	16:50	35 784	23 292	12 492				

一般的,预期系统负载 Cb_t 比系统实际负载 C_t 小,但当系统总的行为数变化量较大时,预期系统负载 Cb_t 比系统实际负载 C_t 大.

例如,当 $Time=14:00$ 时,

$C_t=B_t\times a=97\,274\times 1=97\,274$,

$\Delta S_t=48\,497-48\,324=173$,

$\Delta Br_t=48\,777-41\,598=7179$,

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 1.02$,

$Cb_t=NTR\times Bf_t\times a\approx 1.02\times 48\,497\times 1$

$\approx 49\,467<C_t=97\,274$;

当 $Time=14:10$ 时,

$C_t=B_t\times a=90\,288\times 1=90\,288$,

$\Delta S_t=52\,482-48\,497=3985$,

$\Delta Br_t=48\,777-37\,806=10\,971$,

① 12306 流量超京东商城. http://news.youth.cn/jsxw/201301/t20130122_2825857.htm, 2013,1,22

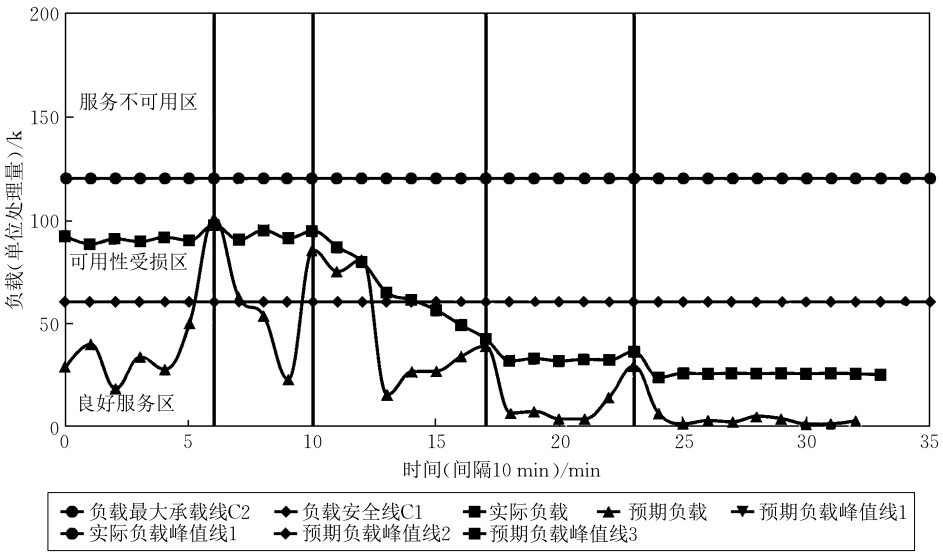

图 9　第 1 组实验结果

$$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 1.92,$$

$$Cb_t=NTR\times Bf_t\times a\approx 1.92\times 52482\times 1\approx 100765>C_t=90288.$$

总之，该组数据中的系统用户量 S_t 和用户重复提交的行为数 Br_t 的变化量大都趋于平稳，此时 S_t 和 Br_t 的变化量对系统负载的影响比较小，导致系统异常感知模型中的放大因子 NTR 的值不会变大，根据放大因子和系统预期负载的关系模型可以得到放大因子 NTR 的值不会使系统预期负载 Cb_t 的值大于系统最大承受负载 $C_{\max}$，即系统不会发出预警信号.

第 2 组实验数据见表 6，把表 6 的数据运用到感知检测算法中，得到效果如图 10 所示. 从图 10 可

表 6　第 2 组实验数据

Time	B_t	Bf_t/S_t	Br_t	*Time*	B_t	Bf_t/S_t	Br_t	*Time*	B_t	Bf_t/S_t	Br_t
13:00	19812	10882	8930	14:40	73741	40629	33112	16:20	48192	30744	17448
13:10	25308	19044	6264	14:50	64474	38543	25931	16:30	37050	25716	11334
13:20	42018	30114	11904	15:00	52272	31896	20376	16:40	47340	29712	17628
13:30	49712	37315	12397	15:10	36570	23892	12678	16:50	32537	20456	12081
13:40	60060	43512	16548	15:20	37248	25134	12114	17:00	27984	18312	9672
13:50	88068	49548	38520	15:30	36024	24108	11916	17:10	25074	18888	6186
14:00	65922	42816	23106	15:40	43696	28170	15528	17:20	25476	18582	6894
14:10	72936	47052	25884	15:50	50486	30982	19504	17:30	26208	19356	6852
14:20	80184	45987	34197	16:00	55424	36384	19040	17:40	25254	18984	6270
14:30	91384	48345	43039	16:10	70716	42072	28644				

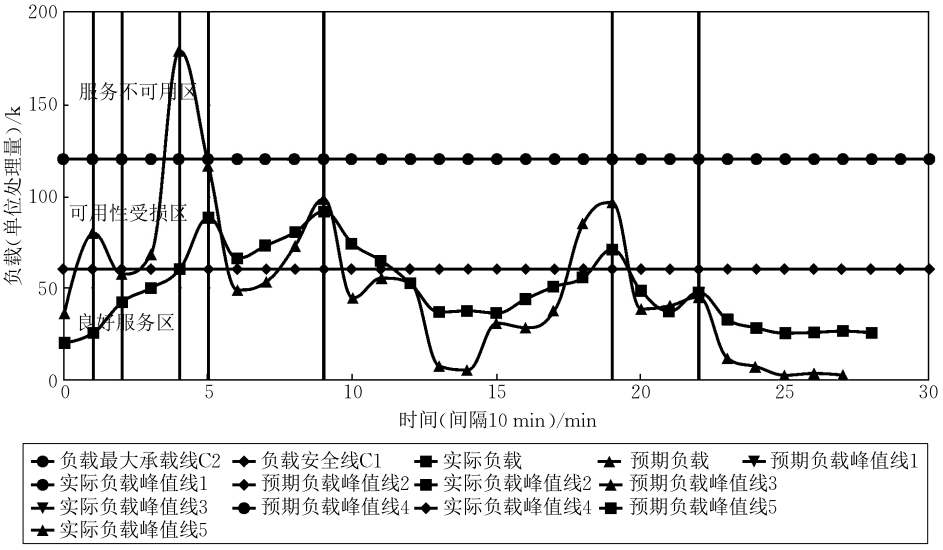

图 10　第 2 组实验结果

以看出，系统实际负载随着时间的推移逐渐上升，且系统预期负载随着实际负载的变化量增长而变大.

例如，当 $Time=15:40$，

$C_t=B_t\times a=43\,696\times 1=43\,696$，

$\Delta S_t=28\,170-24\,108=4062$，

$\Delta Br_t=15\,526-11\,916=3610$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 1.08$，

$Cb_t=NTR\times Bf_t\times a\approx 1.08\times 28\,170\times 1$

$\approx 30\,528<C_t=43\,696$；

当 $Time=15:50$，

$C_t=B_t\times a=50\,486\times 1=50\,486$，

$\Delta S_t=30\,982-28\,170=2812$，

$\Delta Br_t=19\,504-15\,526=3978$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 0.91$，

$Cb_t=NTR\times Bf_t\times a\approx 0.91\times 30\,982\times 1$

$\approx 28\,194<C_t=50\,486$；

当 $Time=16:00$ 时，

$C_t=B_t\times a=55\,424\times 1=55\,424$，

$\Delta S_t=36\,384-30\,982=5402$，

$\Delta Br_t=19\,504-19\,040=464$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 1.03$，

$Cb_t=NTR\times Bf_t\times a\approx 1.03\times 36\,384\times 1$

$\approx 37\,476<C_t=55\,424$；

当 $Time=16:10$ 时，

$C_t=B_t\times a=70\,716\times 1=70\,716$，

$\Delta S_t=42\,072-36\,384=5688$，

$\Delta Br_t=28\,644-19\,040=9604$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 2.02$，

$Cb_t=NTR\times Bf_t\times a\approx 2.02\times 42\,072\times 1$

$\approx 84\,986>C_t=70\,716$.

总之，该组数据中系统用户量 S_t 和用户重复提交的行为数 Br_t 缓慢变化. 随着 S_t 和 Br_t 的缓慢增加，感知模型中放大因子 NTR 也随之缓慢增大，此时系统有发生异常的可能性. 当预期系统负载值 Cb_t 超过系统实际负载 C_t 并先到达系统所能承受的最大负载 $C_{\max}$，系统发出预警信号.

第 3 组实验数据见表 7，把表 7 的数据运用到感知检测算法中，得到效果如图 11 所示. 从图 11 可

表 7 第 3 组实验数据

Time	B_t	Bf_t/S_t	Br_t	Time	B_t	Bf_t/S_t	Br_t	Time	B_t	Bf_t/S_t	Br_t
13:00	45062	28264	16798	15:10	64702	32536	32166	17:20	139572	79829	59743
13:10	44570	28576	15994	15:20	86837	44579	42258	17:30	122156	68862	53294
13:20	44006	28090	15916	15:30	105946	54042	51904	17:40	115406	55478	59928
13:30	43256	27922	15334	15:40	107248	50134	57114	17:50	106762	52110	54652
13:40	62325	36293	26032	15:50	96024	44108	51916	18:00	95856	52584	43272
13:50	85898	48536	37362	16:00	77698	45170	32528	18:10	88692	50790	37902
14:00	70042	40512	29530	16:10	67524	39384	28140	18:20	83526	48096	35430
14:10	59204	30294	28910	16:20	50924	31922	19002	18:30	77176	43352	33824
14:20	42858	28422	14436	16:30	74188	40382	33806	18:40	65182	41876	23306
14:30	34590	23892	10698	16:40	96528	51054	45474	18:50	50136	38747	11389
14:40	36084	22974	13110	16:50	105670	55256	50414	19:00	48390	35874	12516
14:50	28752	17448	11304	17:00	126732	60108	66624	19:10	46105	32997	13108
15:00	57413	29257	28156	17:10	130342	64137	66205	19:20	42379	29794	12585

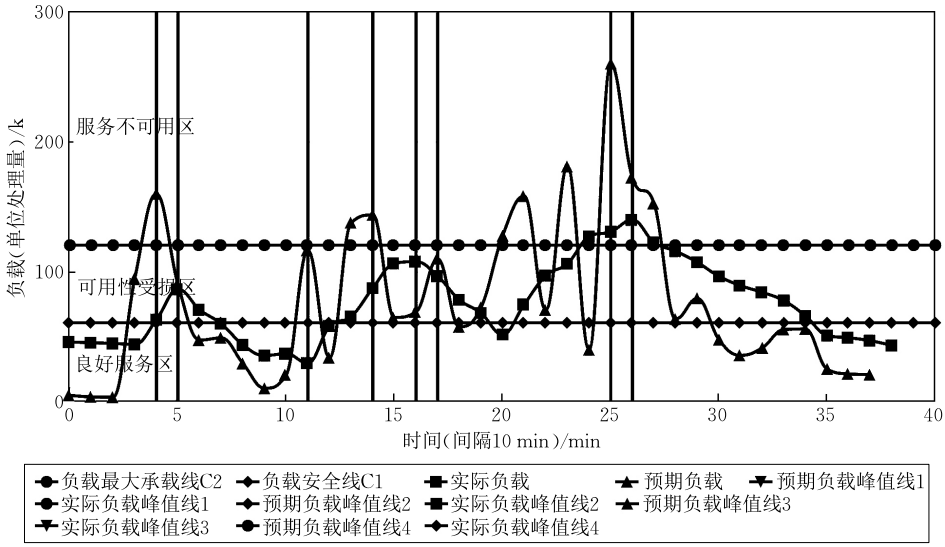

图 11 第 3 组实验结果

以看出，系统实际负载随着时间的推移瞬间上升，且系统预期负载也随着实际负载的变化快速增大.

例如，当 $Time=13:30$ 时，

$C_t=B_t\times a=43\,256\times 1=43\,256$，

$\Delta S_t=28\,090-27\,922=168$，

$\Delta Br_t=15\,916-15\,334=582$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 0.10$，

$Cb_t=NTR\times Bf_t\times a\approx 0.10\times 27\,922\times 1$

$\approx 2792<C_t=43\,256$；

当 $Time=13:40$，

$C_t=B_t\times a=62\,325\times 1=62\,325$，

$\Delta S_t=36\,293-27\,922=8371$，

$\Delta Br_t=26\,032-15\,334=10\,698$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 2.60$，

$Cb_t=NTR\times Bf_t\times a\approx 2.60\times 36\,293\times 1$

$\approx 94\,190>C_t=62\,325$；

当 $Time=13:50$，

$C_t=B_t\times a=85\,898\times 1=85\,898$，

$\Delta S_t=48\,536-36\,293=12\,243$，

$\Delta Br_t=37\,362-26\,032=11\,330$，

$NTR=\beta_1\Delta Br_t/\Delta t+\beta_2\Delta S_t/\Delta t\approx 3.29$，

$Cb_t=NTR\times Bf_t\times a\approx 3.29\times 48\,536\times 1$

$\approx 159\,683>C_{\max}=12\,000>C_t=85\,898$.

总之，该组数据中系统用户量 S_t 和用户重复提交的行为数 Br_t 短时聚集，此时放大因子 NTR 迅速增大，且预期系统负载值 Cb_t 也快速超过系统实际负载 C_t 先到达系统所能承受的最大负载 $C_{\max}$，系统发出预警信号.

通过这 3 组实验的效果图可以看出，如果系统会发生异常，预期负载值总是先于系统实际负载值到达系统所能承受的最大负载值，并进入系统的服务受损区或不可用区. 该实验表明，感知检测算法可以把系统实际负载微小的变化通过放大因子观察预期负载变化，可以有效预警系统可能发生的异常. 同时表明，若系统因行为数的较大变化而发生异常，该算法一定可以提前快速感知.

6　相关讨论

通过环境上下文获取和处理软件的环境信息[7-15]，这种方法可能会导致出现新的上下文，软件就需停止运行，需要较大的系统代价；对于 Web 用户访问行为的这些感知异常行为的方法[16-17]，主要是针对各种异常行为施加到系统上，没有考虑用户合法行为对系统造成的影响；基于流量来检测系统异常[18-20]，只是从表征出发，没有实际考虑系统负载问题；而软件系统的故障检测[21-26]基本上都是基于数据挖掘的角度提出的检测方法，用采集的数据来分析系统是否发生异常，计算量较大，缺乏实时性. 由于大规模网络服务系统的开放性特征，使得不可预见的用户的需求及其行为对系统的可控性提出了更高的要求. 而系统的实时可控是以实时可预测系统异常为前提的. 针对施加于系统的非法行为，现有的感知和检测方法取得了积极效果；但是对于施加于系统的群体合法行为（大规模用户短时聚集并施加于系统的行为，每个用户的行为都是系统允许的）而造成系统异常的情况，目前未见有实时感知技术的研究.

在大规模网络服务系统中，对于群体合法行为的短时聚集而引起系统的异常，本文的敏捷感知方法能以系统负载的细微变化而快速预测系统是否可能发生异常. 即系统若发生异常，本文的方法一定能提前检测到. 这点由第 3 节的定理所保证. 在实际系统中，本文第 4 节给出了一种系统异常检测实现技术，其包括对实际系统的关键行为上施加相应的检测控制. 这种控制通过实时监测系统的行为数，应用感知模型计算实际负载和预期负载. 当预期负载大于实际负载时，系统可能发生异常；此时，通过计算出行为阻滞度来定位可能发生异常的系统行为. 第 5 节的实验表明当预期负载大于实际负载时，系统会发出异常预警.

7　结　论

本文针对大规模网络服务系统环境中用户合法行为数增长而使系统发生异常问题，提出了一种系统异常敏捷感知模型和基于带优先关系的颜色双变迁 Petri 网的重复行为数检测模型及算法. 异常敏捷感知模型是通过行为与行为之间的影响因素，把当前的系统负载在“放大因子”的作用下放大来看，在系统发生异常之前，敏捷感知系统是否会发生异常. 异常敏捷感知模型为实际系统异常发生是否能提前敏捷感知的依据. 在实际系统中可以应用基于带优先关系的颜色双变迁 Petri 网的重复行为数检测模型及算法检测系统的潜在异常发生的可能性，其关键是在实际系统的关键行为节点上增加相应的行为控制部分（图 6 所示），并运用感知模型进行计算预期负载是否大于系统最大负载. 下一步我们将

在感知到导致系统发生异常行为后如何进行系统服务流程重构进行研究和讨论.

参 考 文 献

[1] Ding Bo. The Software Adaptive Several Key Technology Research [Ph. D. dissertation]. National University of Defense Technology, Changsha, 2010(in Chinese)
(丁博. 软件自适应若干关键技术研究[博士学位论文]. 国防科学技术大学, 长沙, 2010)

[2] Bao Xiao-An, Yao Lan, Zhang Na, et al. Adaptive software testing based on controlled Markov chain. Journal of Computer Research and Development, 2012, 49(6): 1332-1338(in Chinese)
(包晓安, 姚澜, 张娜等. 基于受控 Markov 链的软件自适应测试策略. 计算机研究与发展, 2012, 49(6): 1332-1338)

[3] Jie Jun-Jing, Shi Ting-Xun, Jiao Wen-Pin, et al. Autonomous components whose adaptation policies can be customized online and evaluated dynamically. Journal of Software, 2012, 23(4): 802-815(in Chinese)
(接钧靖, 史庭训, 焦文品等. 自主构件自适应策略的在线定制及动态评估. 软件学报, 2012, 23(4): 802-815)

[4] Ehlers J, van Hoorn A, Waller J, et al. Self-adaptive software system monitoring for performance anomaly localization//Proceedings of the 8th ACM International Conference on Autonomic Computing. Karlsruhe, Germany, 2011: 197-200

[5] Ehlers J, Hasselbring W. Self-adaptive software performance monitoring//Proceedings of the Software Engineering. Karlsruhe, Germany, 2011: 51-62

[6] Ding Bo, Wang Huai-Min, Shi Dian-Xi. Constructing software with self-adaptability. Journal of Software, 2013, 24(9): 1981-2000(in Chinese)
(丁博, 王怀民, 史殿习. 构造具备自适应能力的软件. 软件学报, 2013, 24(9): 1981-2000)

[7] Ding Bo, Wang Huai-Min, Shi Dian-Xi. Middleware technology in pervasive computing. Journal of Frontiers of Computer Science and Technology, 2007, 1(3): 241-254(in Chinese)
(丁博, 王怀民, 史殿习. 普适计算中间件技术. 计算机科学与探索, 2007, 1(3): 241-254)

[8] Zhu Ke-Ke. Research on the Adaptive Middleware Technology for Environmental Perception [Ph. D. dissertation]. Wuhan University of Technology, Wuhan, 2013(in Chinese)
(朱可可. 面向环境感知的自适应中间件技术的研究[博士学位论文]. 武汉理工大学, 武汉, 2013)

[9] Zhang Hui. Research on Context Aware System for Pragmatic Web [Ph. D. dissertation]. Dalian Maritime University, Dalian, 2015(in Chinese)
(张慧. 面向语用网的上下文感知系统研究[博士学位论文]. 大连海事大学, 大连, 2015)

[10] Chen G, Kotz D. A survey of context-aware mobile computing research. Department of Computer Science, Dartmouth College: Technical Report TR2000-381, 2000

[11] Yang Y, Huang Y, Cao J, et al. Formal specification and runtime detection of dynamic properties in asynchronous pervasive computing environments. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(8): 1546-1555

[12] Cheng S W. Rainbow: Cost-effective software architecture-based self-adaptation. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA: Technical Report CMU-ISR-08-113, 2008

[13] Robinson W N. Monitoring web service requirements//Proceedings of the 11th IEEE International Requirements Engineering Conference. Monterey Bay, USA, 2003: 65-74

[14] Garlan D, Schmerl B, Chang J. Using gauges for architecture-based monitoring and adaptation//Proceedings of the Working Conference on Complex and Dynamic Systems Architecture. Brisbane, Australia, 2001: 12-14

[15] Mi H, Wang H, Yin G, et al. Magnifier: Online detection of performance problems in large-scale cloud computing systems //Proceedings of the 8th IEEE 2011 International Conference on Services Computing (SCC). Washington, USA, 2011: 418-425

[16] Tian Xin-Guang, Gao Li-Zhi, Sun Chun-Lai, et al. Anomaly detection of program behaviors based on system calls and homogeneous Markov chain models. Journal of Computer Research and Development, 2007, 44(9): 1538-1544 (in Chinese)
(田新广, 高立志, 孙春来等. 基于系统调用和齐次 Markov 链模型的程序行为异常检测. 计算机研究与发展, 2007, 44(9): 1538-1544)

[17] Xie Yi, Yu Shun-Zheng. Statistical anomaly detection based on Web user's browsing behavior. Journal of Software, 2007, 18(4): 967-977(in Chinese)
(谢逸, 余顺争. 基于 Web 用户浏览行为的统计异常检测. 软件学报, 2007, 18(4): 967-977)

[18] Feng Zhen. The Correlation Analysis of Backbone Network Traffic Anomaly Events [Ph. D. dissertation]. University of Electronic Science and Technology, Chengdu, 2010 (in Chinese)
(冯震. 骨干通信网中流量异常事件的关联分析[博士学位论文]. 电子科技大学, 成都, 2010)

[19] Huang T, Ma X, Ji X, et al. An efficient method for online detecting abnormal cascading pattern in distributed networks //Proceedings of the 10th International Conference on Fuzzy Systems and Knowledge Discovery(FSKD). Shenyang, China, 2013: 741-746

[20] Li Z L, Hu G M, Yang D. Global abnormal correlation analysis for DDoS attack detection//Proceedings of the 2008 IEEE Symposium on Computers and Communications. Marrakech, Morocco, 2008: 310-315

[21] Shan Jin-Hui, Xu Ke-Jun, Wang Ji. A kind of software faults diagnosing framework. Chinese Journal of Computers, 2011, 34(2): 372-373(in Chinese)
(单锦辉，徐克俊，王戟. 一种软件故障诊断过程框架. 计算机学报，2011，34(2)：372-373)

[22] Smith D, Guan Q, Fu S. An anomaly detection framework for autonomic management of compute cloud systems//Proceedings of the 2010 IEEE 34th Annual Computer Software and Applications Conference Workshops (COMPSACW). Seoul, Korea, 2010: 376-381

[23] Bertier M, Marin O, Sens P. Performance analysis of a hierarchical failure detector//Proceedings of the International Conference on Dependable Systems and Networks. San Francisco, USA, 2003: 635-644

[24] Shi X, Jin H, Han Z, et al. ALTER: Adaptive failure detection services for grids//Proceedings of the 2005 IEEE International Conference on Services Computing. Orlando, USA, 2005: 355-358

[25] Wang C, Talwar V, Schwan K, et al. Online detection of utility cloud anomalies using metric distributions//Proceedings of the 12th IEEE/IFIP Network Operations and Management Symposium (NOMS). Osaka, Japan, 2010: 96-103

[26] Basile F, Cabasino M P, Seatzu C. State estimation and fault diagnosis of labeled time petri net systems with unobservable transitions. IEEE Transactions on Automatic Control, 2015, 60(4): 997-1009

[27] Wu Zhe-Hui. Petri Net Introduction Theory. Beijing: China Machine Press, 2006(in Chinese)
(吴哲辉. Petri 网导论. 北京：机械工业出版社，2006)

[28] Gao T, Li T, Xie Z, et al. A process model of software evolution requirement based on feedback//Proceedings of the 2011 International Conference on Information Technology, Computer Engineering and Management Sciences (ICM). Kunming, China, 2011: 171-174

[29] Best E, Koutny M. Petri net semantics of priority systems. Theoretical Computer Science, 1992, 96(1): 175-215

[30] Sheng You-Zhao. Queuing Theory and Its Application in Modern Communications. Beijing: The People' Posts and Telecommunications Press, 2007(in Chinese)
(盛友招. 排队论及其在现代通信中的应用. 北京：人民邮电出版社，2007)

[31] Lin Chuang, Tian Li-Qin, Wei Ya-Ya. Performance equivalent analysis of workflow systems. Chinese Journal of Computers, 2002, 13(8): 1472-1480(in Chinese)
(林闯，田立勤，魏丫丫. 工作流系统模型的性能等价分析. 计算机学报，2002，13(8)：1472-1480)

ZHANG Zhao-Hui, born in 1971, Ph. D., professor. His main research interests include Internet information service, service computing and cloud computing.

CUI Jun, born in 1992, M. S. candidate. Her research interests include clouding computing and service computing.

Background

Compared with the traditional computer software systems, the service systems based on Internet show the characteristics of large scale, openness, complexity and so on. In a large-scale network service system, the instantaneous gathering of a number of legal users can cause the abnormality of system behaviors and vastly damage the system availability. But the most of anomaly detection methods aim to the illegal user behavior. And these methods cannot effectively support agile perception for abnormal behaviors while the system is normal.

In this paper, we define the Magnification Factor to magnify the tiny changes of the system load. Based on it, an agile perception model for the system anomalies is presented. By the model we conclude that the predictability of system anomalies and the localizability of abnormal system behaviors. Furthermore, based on the model, we present a Petri Net detection model with priority relation and color double transition.

This research was supported by the National Natural Science Foundation of China (No. 61472004). It is a critical challenge of technologies and theories on usability of the large-scale network service systems that system behavior is adapted softly to keep the good system service state with the fixed resources capacity and disturbance. There are two key problems to be solved: one is atomic perception of group behavior variation and another is adaptive behavior reconstruction under load balance. All research results are valuable to technologies and theories on usability, and Petri Nets. The results can support agile variant behavior detecting and workflow performance adapting for the large-scale network service systems. This paper has resolved the agile perception of system anomalies caused by legal behaviors of large scale users. It is just a part of the first problem of the project.