一种面向智能网络系统的自主计算能力分析方法*

孙日明，胡先浪
（江苏自动化研究所，江苏 连云港 222061）

摘 要：目前各类智能网络系统已被广泛地应用，但是由于节点众多、外部环境复杂，自我管理具有很大挑战。而自主计算系统(ACS)具有根据策略和目标实现自主管理的能力，在复杂的智能网络系统中具有广阔的应用前景。然而，目前自主计算的评价方法缺乏准确的量化来评估 ACS 的自我管理水平。首先提出了基于 PEPA(Performance Evaluation Process Algbra)的自主计算评价模型。然后，根据自主计算的核心思想(较少或无人干预)提出了一种自我管理的评价指标。此外，为了避免 ACS 的巨大规模导致传统马尔可夫链的状态空间爆炸，采用连续状态空间近似方法从 PEPA 模型中生成 ODEs(Ordinary Differential Equations)。实验结果表明，提高检测成功率和 self-* 变迁速率对提高自主计算具有重要意义，为自主计算提供了一种评价方法，可以自动测量自我管理的能力。

关键词：自主计算；PEPA；连续状态空间近似
中图分类号：TN919.5；TP393 **文献标识码**：A DOI：10.16157/j.issn.0258-7998.200811

中文引用格式：孙日明，胡先浪. 一种面向智能网络系统的自主计算能力分析方法[J].电子技术应用，2021，47(9)：59-63，68.
英文引用格式：Sun Riming，Hu Xianlang. An analysis method of autonomic computing capability for intelligent network system[J]. Application of Electronic Technique，2021，47(9)：59-63，68.

An analysis method of autonomic computing capability for intelligent network system

Sun Riming，Hu Xianlang
（Jiangsu Automation Research Instisute，Lianyungang 222061，China）

Abstract：At present，all kinds of intelligent network systems have been widely used，but due to the large number of nodes and complex external environment，its self-management has great challenges. The autonomous computing system(ACS) has the ability to manage itself according to the strategy and goal，and has broad application prospects in the complex intelligent network system. However，the current evaluation method of autonomic computing lacks accurate quantification to evaluate the self-management level of ACS. This paper firstly proposes an evaluation model of autonomic computing based on PEPA(performance evaluation process algbra). Then，according to the core idea of autonomic computing(less or no intervention)，a self-management evaluation index is proposed. In addition，in order to avoid the state space explosion of traditional Markov chain caused by the huge scale of ACS，ODEs (ordinary differential equations) are generated from PEPA model by using continuous state space approximation method. The experimental results show that improving the detection success rate and self-* transition rate is of great significance to improve autonomic computing. This work provides an evaluation method for autonomic computing，which can automatically measure the self-management ability.

Key words：autonomic computing；PEPA；continuous state space approximation

0 引言

随着人工智能、云计算和边缘计算的发展，网络的智能化水平越来越高。涌现了车联网、自组织传感器网络等一系列智能网络系统[1]，在国民经济、国防安全等领域发挥了重要作用。然而这些系统往往具有大量的节点且节点均具有较高的智能化水平，如何实现智能网络系统的自我管理(是指能够根据环境的变化自动地调整，以满足各种需求)已成为一项巨大的挑战[2]。

为了以合理的成本应对分布式系统的管理复杂性，IBM 在 2001 年提出了自主计算能力[3]。其目的是能够感知和自我管理，并以最少的人为干预来处理复杂性和不确定性，被广泛应用于航天控制、智能交通等领域。目前，自主计算也已被认为是智能网络管理的一种有效技术途径，是智能网络系统的一种基础能力。虽然自主计算体系结构和设计方法领域已经取得了一些显著的成果，但自主计算的评价仍处于早期阶段。对自我管理评价的研究有助于发现智能网络系统的缺陷，为进一步的设计提供有效的参考。

* 基金项目：国家重点研发计划项目(2018YFB1305900)

到目前为止,一些学者已经为自主计算系统提出了一些评估指标、模型和简单的工具,但这些研究仍然集中在某些特定属性和基于规则的阶段,缺乏系统的评估模型和复杂的测量工具。本文首先简单地总结了未解决的问题和挑战,然后提出了一种基于 PEPA 的自主计算能力分析模型。在此基础上,本文将模型转换为 ODEs(Ordinary Differential Equations),用自主指标的度量来分析自我管理的能力,并给出了结论。

1 相关工作

自主计算通常由其主要属性定义,包括自配置、自愈、自优化和自我保护等。自主计算系统的目的是通过其管理员对给定的高层目标进行自我管理,隐藏系统的复杂性。目前,自主计算评价的研究还处于早期阶段,存在一些不足。

(1)目前,关于自主计算能力的评价研究大多集中在建立度量集。McCann 等人[4]提出了一套包括 10 个度量标准的自主计算度量标准;Kaddoum 等人[5]对自主计算的 self-* 属性进行评价,但是复杂性较高。Janeska 等人[6]从包括需求在内的多个侧面构建了一个自主管理系统评估度量体系。但是,所有这些度量或度量集都不能评估自主计算的所有特征,其中大多数缺乏定量方法,不能满足细粒度评估的要求。

(2)自主计算目前只有 3 种评价模型。其中两个是 IBM 提出的,分别是自主成熟度模型和自主适应模型。在前者中,自主计算系统分为 5 个层次:基础性、管理、预测性、适应性和自主性。这种分类的粒度太粗糙,无法进行严格的评价。在后者中,自主化的能力是通过功能、控制范围和服务流程等方面来衡量的,但由于其粗糙粒度,只能得到定性的结果。第 3 种评价模型主要采用了 AHP(Analysis Hierarchy Process)方法,文献[7]中基于 AHP 对自主计算能力进行分析,而 Khorsand 等人[8]提出一种基于 FAHP(Fuzzy AHP)的方法用于分析云环境下自主提供服务的能力,其中度量的权重系数具有很大的主观性,难以客观量化。

(3)现有的自主计算评估工具是 IBM 自动化评估工具,它通过用户提交的表格来测量被测试的系统。这个工具的粒度太粗糙,无法在大规模的智能网络系统中使用。并且其他的所有方法都缺乏自动评价工具,效率较低。

(4)此外,另一些学者也结合具体业务系统对自主计算的评价进行了专门研究。Sanchez 等人[9]结合工业 4.0 网络系统的自主化管理,研究了工业过程(包括数据、人员、事物和服务)实现 self-* 的评价准则,但是与工业过程紧密耦合,迁移性不强。文献[10]重点分析了现存大型分布式系统的自主计算能力,并对如何选择配置参数等方面进行了分析。Jaleel 等人[11]分别针对云环境和普通计算环境提出了一系列的指标(例如数据通道需求、对现有业务的扰动)来评价系统的自主管理能力和 self-* 能力。Singh 等人[12]也针对云计算系统下计算资源的自主管理能力进行了评价,但是这种评价主要针对 QoS(Quality of Service)应用范围还有一定局限性。

从以上讨论可以看出,ACS 的评价需要进一步的研究,才能得到一种细粒度的定量方法。因此,本文将建立一个 ACS 评价模型,以支持定量分析和评估。

2 PEPA 的连续状态空间近似

PEPA 是一种经典的进程代数,具语义验证和定量分析能力,适用于大规模分布系统的建模和分析。PEPA 的语法如下:

$$P ::= (a,\lambda).P \mid P+P \mid P \underset{L}{\bowtie} P \mid P/L \mid A \tag{1}$$

其中,λ 是动作 a 的变迁速率,其他操作还包括选择+、合作 $\underset{L}{\bowtie}$、隐藏/以及常量定义 A。详情可参考文献[13]。

传统上,PEPA 通常被映射到一个底层的马尔可夫链来分析性能。然而,当存在大的状态空间时,通常会遇到状态空间爆炸。可以采用连续状态空间近似方法将马尔可夫链生成 ODEs[10]来解决这个问题。

设 $N(C_{ij},t)=v_{ij}(t)$ 表示在时刻 t 时第 i 个子向量的第 j 个入口。用 $\mathrm{Exit}(C_{ij})$ 表示从 $N(C_{ij},t)$ 发出动作的集合,$\mathrm{Exit}(C_{ij})$ 表示进入 $N(C_{ij},t)$ 动作的集合。PEPA 模型可转化为:

$$\frac{\mathrm{d}v_{ij}(t)}{\mathrm{d}t}=-\sum_{\alpha\in \mathrm{Exit}(C_{ij})}\rho_{\alpha}(C_{ij},P(t))+\sum_{\beta\in \mathrm{Enter}(C_{ij})}\rho_{\beta}(C_{ik},P(t))$$
$$i=1,2,\cdots,n;\ j=1,2,\cdots,N_i \tag{2}$$

其中,转移概率 $\rho_{\alpha}(C_{ij},P(t))$ 代表在系统 $P(t)$ 中当动作 α 发生时,C_{ij} 组件减少的概率;$\rho_{\beta}(C_{ik},P(t))$ 代表在系统 $P(t)$ 中当动作 β 发生时,C_{ik} 组件减少的概率。

组件的稳态似然概率等于马尔可夫链计算的稳态概率,其行向量表示为 $\pi=\{\pi_1,\pi_2,\cdots\}$[14]。

3 自主计算能力的评价模型

3.1 评价模型

在本质含义上,自主计算是以最小或无人干扰的自我管理方式为用户服务,换言之,自主计算与服务过程密切相关。因此,ACS 可以从服务的角度进行研究,状态代表了自我管理过程中的不同阶段。根据自主计算的特点和服务过程,ACS 的状态集表示为 $S=\{G,V,A,SC,SH,SO,SP,F,L\}$,各状态如下:

(1)G(General State):系统处于正常服务状态;

(2)V(Vulnerability State):该系统容易受到可疑服务要求或恶意攻击;

(3)A(Adaptation State):检测该系统以确定是否采取自我管理行动;

(4)SC(Self-Configuration State):开展自我配置活动;

(5)SH(Self-Healing State):实施自愈活动;

(6)SO(Self-Optimization State):开展自我优化活动;

(7)SP(Self-Protection):开展自保护活动;

(8)F(Failure State):不能自主管理该系统,进入失效状态;

(9)L(Learning State):通过自我学习,升级应对故障。

这些状态之间的关系如图 1 所示。在一开始,系统停留在 G 状态。在遭受一些可疑服务需求或恶意攻击时,以一定的概率转向 V 状态。然后,系统将确定它是否可以使用当前的自我管理措施进行处理。如果答案是肯定的,则系统进入自适应状态(Adaptation State),并开始实现 self-* 活动;否则系统进入 F 状态。虽然采取的 self-* 措施是有效的,但该系统仍需要回到 G 状态,否则它再次达到适应状态。在适应状态下,如果所有的 self-* 措施都不能实现自我管理,系统也会转到 F 状态。由于未能自行管理系统,采用自主学习的过程重新生成自主系统的知识库,然后系统回到 G 状态。

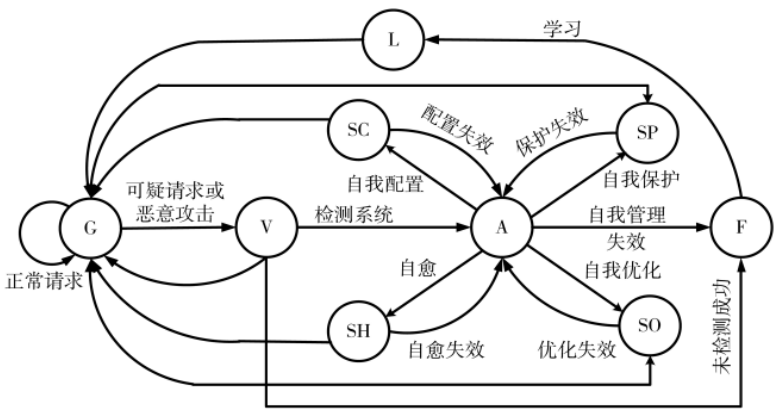

图 1 自主系统的状态转移图

根据图 1 所示,从服务的角度出发,采用 PEPA 对自主计算系统建模。所有的服务请求都可以看作是一系列的进程,具体描述如下:

Client:=(send_request,$v1$).Waiting;

Waiting:=(wait,$v2$).Response;

Response:=(serve,$v3$).waiting+(finish,$v4$).Client+
(refuse,$v5$).Client;

相应地,自主计算系统建模如图 2 所示,其中所有动作的延迟时间都遵循负指数分布。所有的动作速率都可以用 Huang 的方法得到[15]。

General:=(serve,τ). Vulnerable+(serve,τ).General;
Vulnerable:=(mask,c).General+(adapt,b).Adaptation+
(undetected,c).Fail;
Adaptation:=(self_configure,$r1$).SelfConfigurition+
(self_protecte,$r2$).SelfProtection+(self_healing,$r3$).SelfHealing+
(self_optimize,$r4$).SelfOptimization+(fail,f).Fail;
SelfConfigurition=(configure,$w1$).General+(configure_fail,$w2$).Adaptation;
SelfProtection=(protect,$w3$).General+(protect_fail,$w4$).Adaptation;
SelfHealing=(heal,$w5$).General+(heal_fail,$w6$).Adaptation;
SelfOptimization=(optimize,$w7$).General+(optimize_fail,$w8$).Vulnerable;
Fail=(recover,$w9$).Learn;
Learn=(learn,l).General;

图 2 基于 PEPA 的自主计算系统模型

综上,通过服务交互,自主计算系统模型可以描述为:

$$\text{Client}[N] \underset{\text{serve}}{\triangleright\!\triangleleft} \text{General} \tag{3}$$

其中,N 表示用户数,$\underset{\text{serve}}{\triangleright\!\triangleleft}$代表以 serve 动作协作。

3.2 转化为 ODEs 求解

在本节中,将 PEPA 模型转换为 ODEs 进行求解,以避免状态爆炸。

首先需要为系统创建一个基于时间的系统方程,然后才能使用 ODEs 生成方法。令:

$$\boldsymbol{P}_1(t)=(N(\text{Client},t),N(\text{Waiting},t),N(\text{Plan},t),N(\text{Response},t)) \tag{4}$$

$$\boldsymbol{P}_2(t)=(N(\text{G},t),N(\text{V},t),N(\text{A},t),N(\text{SC},t),N(\text{SP},t),N(\text{SH},t),N(\text{SO},t),N(\text{F},t),N(\text{L},t)) \tag{5}$$

其中,$N(\text{C},t)$表示时间 t 中组件 C 的个数。可以定义一个基于时间的系统:

$$\boldsymbol{P}(t)=\boldsymbol{P}_1(t)\triangleright\!\triangleleft\boldsymbol{P}_2(t) \tag{6}$$

利用连续状态空间近似方法,可以将评价模型的状态映射为一系列局部派生,具体如下:

$\text{Client}\rightarrow C_{11}$	$\text{Waiting}\rightarrow C_{12}$	$\text{Response}\rightarrow C_{13}$
$\text{G}\rightarrow C_{21}$	$\text{V}\rightarrow C_{22}$	$\text{A}\rightarrow C_{23}$
$\text{SC}\rightarrow C_{24}$	$\text{SP}\rightarrow C_{25}$	$\text{SC}\rightarrow C_{26}$
$\text{SO}\rightarrow C_{27}$	$\text{F}\rightarrow C_{28}$	$\text{L}\rightarrow C_{29}$

模型中各组件的关系如图 3 所示。

根据式(2)生成 ODEs。

$$\begin{cases}
x'_{11}=-v_1\cdot x_{11}+(v_4+v_5)\cdot x_{13}\\
x'_{12}=-v_2\cdot x_{12}+v_1\cdot x_{11}+I_{21}\cdot v_3\cdot x_{13}\\
x'_{13}=-I_{21}\cdot v_3\cdot x_{13}-(v_4+v_5)\cdot x_{13}+v_2\cdot x_{12}\\
x'_{21}=-I_{21}\cdot v_3\cdot x_{13}+w_1\cdot x_{24}+w_3\cdot x_{25}+w_5\cdot x_{26}+w_7\cdot x_{27}+l\cdot x_{29}\\
x'_{22}=-(a+b+c)x_{22}+I_{21}\cdot v_3\cdot x_{13}\\
x'_{23}=-(r_1+r_2+r_3+r_4+f)\cdot x_{23}+w_2\cdot x_{24}+w_4\cdot x_{25}+w_6\cdot x_{26}+w_8\cdot x_{27}\\
x'_{24}=-(w_1+w_2)x_{24}+r_1\cdot x_{23}\\
x'_{25}=-(w_3+w_4)x_{25}+r_2\cdot x_{23}\\
x'_{26}=-(w_5+w_6)x_{26}+r_3\cdot x_{23}\\
x'_{27}=-(w_7+w_8)x_{27}+r_4\cdot x_{23}\\
x'_{28}=-w_9\cdot x_{28}+c\cdot x_{22}+f\cdot x_{23}\\
x'_{29}=-l\cdot x_{29}+w_9\cdot x_{28}
\end{cases} \tag{7}$$

其中,x_{ij} 表示 C_{ij} 的数量,即对于第 i 种组件的第 j 个派生的数量。而 I_{21} 是一个特定的符号函数:

$$I_{21}=\begin{cases}1 & x_{21}(t)>0\\ 0 & x_{21}(t)=0\end{cases} \tag{8}$$

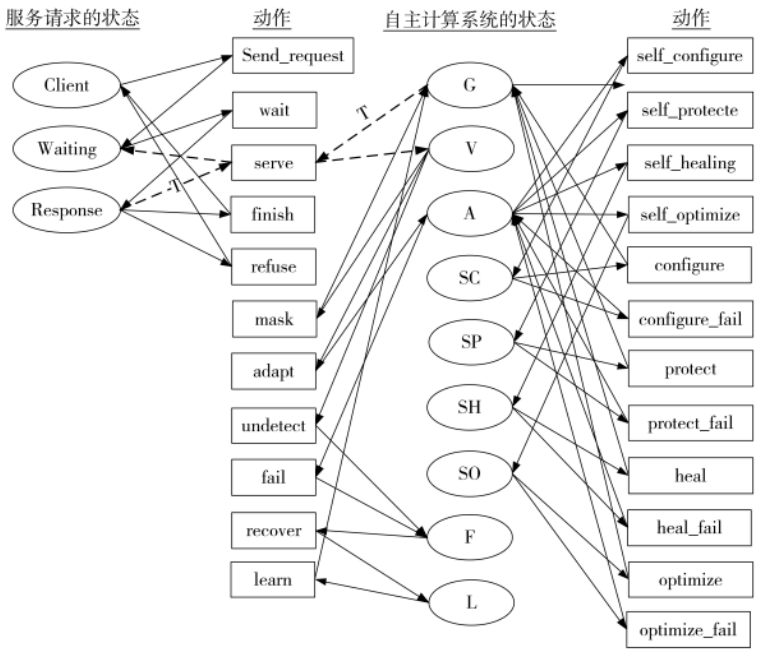

图 3 模型的活动图

4 自主计算的度量

自主计算的显著特点是自我管理，但现有的多属性评价方法和注入实验都没有涵盖影响自我管理的所有因素。本文依据自主计算的核心思想来表征自我管理的状态，而不是使用指标集来评估行为，即在自我管理条件下不受人为干扰地完成任务的概率。并将此概率表示为自主指数，它是一个实数，属于$(0,1]$。

考虑到模型的隐含随机过程是$\{M(T),t>0\}$，$M(T)$中的每个状态都有一个形式为 Client′ ▷◁ General′，其中 Client′和 General′分别是 Client 和 General 状态的派生。本文将模型的状态空间分为两组：自主集合 S_1 和非自主集合 S_2，S_2 中每一种状态都有一种形式的 Client′ ▷◁ Fail。同样地，稳态$\{\pi_1,\pi_2,\cdots,\pi_n\}$由两部分组成：$\Pi_A$ 和 Π_F。其中 Π_F 是与 S_2 相对应的稳态概率子向量。

定义(自主指数)：给定 $\boldsymbol{\pi}=\{\pi_1,\pi_2,\cdots,\pi_n\}$是系统的稳态概率向量，$\Pi_F$ 是对应于 S_2 的稳态概率子向量，自主指标可定义为 $\Gamma_A\in(0,1]$：

$$\Gamma_A=\sum_{\pi_i\in\Pi_F}\pi_i \tag{9}$$

5 仿真实验分析

根据上一节提出的自主指数，将以量化的方式对自我管理能力进行评估。模型的参数可以根据实际自我管理系统的历史日志得到。假设图 2 中模型的各个参数如表 1 所示。

其次，可以将 ODEs 的方法与传统的 DTMC(Distributed Time Markov Chain)进行比较。其中两种方法的计算时间如表 2 所示。其中，“-”表示在内存为 4 GB 的 PC 中，计算资源不足。由于 ODEs 方法变量是连续的，因此没有状态空间。

从表 2 可以看出，DTMC 方法的计算时间大于 ODEs。

表 1 PEPA 模型的参数值

参数	数值	参数	数值
v_1	0.9	r_2	0.25δ
v_2	0.95	r_3	0.15δ
v_3	0.8	r_4	0.3δ
v_4	0.1	w_1	0.99
v_5	$1-v_3-v_4$	w_2	$1-w_1$
α	0.8	w_3	0.99
β	0.999	w_4	$1-w_3$
a	$1-\alpha$	w_5	0.99
b	$\alpha\times\beta$	w_6	$1-w_5$
c	$\alpha(1-\beta)$	w_7	0.99
f	0.001	w_8	$1-w_7$
δ	$1-f$	w_9	0.05
r_1	0.2δ	l	1.0

此外，由于状态空间爆炸，DTMC 方法需要更多的计算资源，甚至需要计算机的内存。根据式(9)，可以得到当 $N=10$ 时自主指数 $\Gamma_A=0.942$。

表 2 测试系统的计算时间

组件数量 N	DTMC		ODEs
	状态空间中的组件数量(在简化之前/之后)	计算时间/s	计算时间/s
3	243/90	0.016	0.31
5	2 187/189	0.032	0.31
10	531 441/594	0.172	0.31
20	3.138×10^{10}/2 079	3.219	0.31
100	4.638×10^{48}/46 359	-	0.31
200	2.391×10^{96}/182 709	-	0.31

接下来以上述案例为实例，分析这些参数对自主指数的影响。通过分析，可以发现提高自我管理能力的关键因素，为自主计算的进一步研究提供参考。为了简化解决的过程，使用了 PEPA Eclipse 插件[14]工具。

5.1 检测能力对自主指数的影响

当可疑的服务需求或恶意攻击发生时，系统可能会选择适应，否则，它将无法在未被发现的情况下管理自己。在图 4 中，给出了检测概率对自主指数的影响，X 轴是参数 β，表示自主计算系统检测漏洞状态的可能性；Y 轴是自主指数。从图 4 中可以看出，自主指数随着检测可能性的增加而增加。当 β 接近 1.0 时，自主指数有加速增长的趋势。因此，提高检测成功率对提高自主系统的能力具有重要意义。

5.2 自主特征对自主指数的影响

自主计算的特点通常被称为 self-*，包括自我配置、自我优化、自我愈合和自我保护，这对于实现自我管理至关重要。在图 5 中，所有的 self-* 特征对自主指数的影响均在同一图中给出。从这个角度看，自主指数随着 self-* 的增加而增加。值得注意的一点是，如果单个

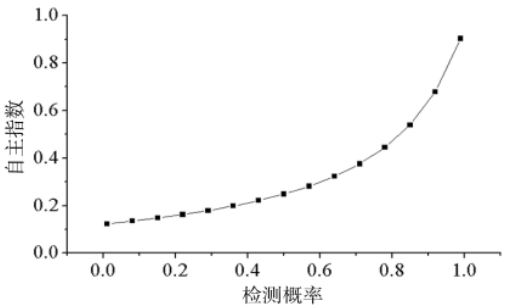

图 4 适应可能性对自主指数的影响

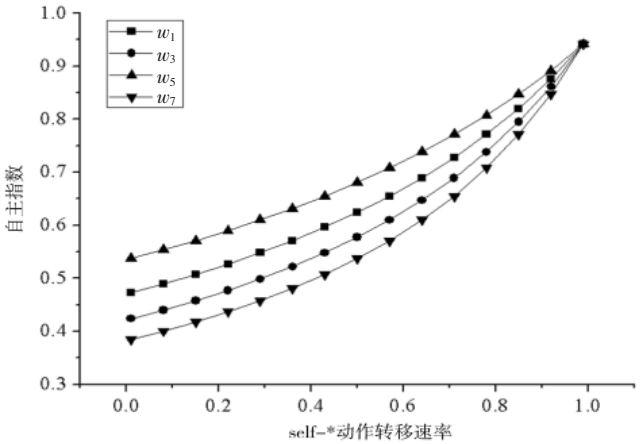

图 5 self-* 率对自主指数的影响

self-* 接近 0，自主指数将保持在一个较低的水平，但不会下降到 0。这种假设与现实相匹配，例如，一个只能自我配置和自我优化的系统尽管自主管理水平不高，但也是一个自主系统，具有自主能力。

5.3 失效率对自主计算能力的影响

有时，自主系统由于知识库的不完备性可能无法自我管理，这对于自主管理能力的影响是致命的。在图 6 中，随着失效率 f 的增长，自主指数也随之下降。此外，当失败率接近 1.0 时，自主指数达到 0.1 以下的值。可以发现，由于故障率的影响是非常严重的，因此应该尽力减少这种可能性。

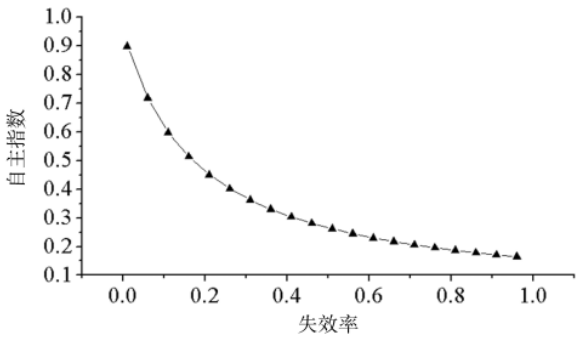

图 6 失效率对自主指数的影响

6 结论

根据自主计算的核心思想，本文提出了一种基于 PEPA 的自主计算能力评估模型，可以用于评估智能网络系统的自主计算能力。与现有的 ACS 评价方法或模型相比，该模型可以通过总结系统自我管理状态的似然稳态概率来进行定量分析。根据 Huang 方法的结果[15]，避免了用户设置一系列的系数，所有的系数都可以通过测量动作平均延迟时间来计算。实验结果表明，提高自主检测成功率和 self-* 率对增强自主能力具有重要意义。

参考文献

[1] GAO Z, ZHANG H, DONG S, et al. Salient object detection in the distributed cloud-edge intelligent network[J]. IEEE Network, 2020, 34(2): 216-224.

[2] DEHRAJ P, SHARMA A. A review on architecture and models for autonomic software systems[J]. The Journal of Supercomputing, 2021, 77(1): 388-417.

[3] MILLARA A F, MOLLEDA J, USAMENTIAGA R, et al. Profile measurement of rails in a rolling mill: implementing and evaluating autonomic computing capabilities[J]. IEEE Transactions on Industry Applications, 2019, 55(5): 5466-5475.

[4] MCCANN J A, HUEBSCHER M C. Evaluation issues in autonomic computing[C]//Proc. Int. Conf. Grid Cooperat. Comput. Berlin, Germany: Springer, 2004: 597 - 608.

[5] KADDOUM E, GLEIZES M P, GEORGE J P, et al. Characterizing and evaluating problem solving self-*systems[C]// Proceedings of the 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns. November 2009: 137-145.

[6] JANESKA M, ZDRAVESKI D, MANCHESKI G, et al. Performance evaluation of autonomic computing systems[C]// 22th International Scientific Conference Strategic Management and Decision Support Systems in Strategic Management, 2017, Nov 19, Subotica, R. Srbija, 2017: 1-12.

[7] Zhang Haitao, Whang Huiqiang, Zheng Ruijuan. An autonomic evaluation model of complex software[C]//2008 International Conference on Internet Computing in Science and Engineering. Harbin, January 28, 2008: 1-6.

[8] KHORSAND R, GHOBAEI-ARANI M, RAMEZANPOUR M. FAHP approach for autonomic resource provisioning of multitier applications in cloud computing environments[J]. Software: Practice and Experience, 2018, 48(12): 2147-2173.

[9] SANCHEZ M, ERNESTO E, JOSE A. Autonomic computing in manufacturing process coordination in industry 4.0 context[J]. Journal of Industrial Information Integration, 2020, 19(9): 100159.

[10] LI H, CHEN T H, HASSAN A E, et al. Adopting autonomic computing capabilities in existing large-scale systems[C]// 2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP). IEEE, 2018: 1-10.

[11] JALEEL A, ARSHAD S, SHOAIB M, et al. Design quality

（下转第 68 页）

(3)科研人员。基于科研人员基础数据,整合多渠道获取的人员科研活动、保障情况、考核业绩等数据,构建科研人员数据集,支撑开展学术力量构成分析、科研人员综合评价分析和科研人员保障规律分析。

(4)科研资源。综合科研方向变化、资源投入规模、资源损耗规律、资源急需程度、资源管理流程等因素,构建科研资源数据集,支撑开展科研资源布局优化、科研资源储备优化和科研资源全寿命管理。

(5)科研趋势。基于各类科研信息服务平台,结合科技发展动态,综合发达国家科研发展情况等,构建科研趋势数据集,支撑开展科研主题统计分析、热点学术问题分析和学科发展走势分析,通过大数据技术手段,分析学科的关注程度、发展趋势、影响程度等。

(6)科研应用。按照科研数据顶层设计和科研领域数据建设应用总体规划,在运用科研数据资源和应用成果基础上,基于领域特色需求和专业科研活动,开展多源汇聚和关联分析,形成与科研体系衔接配套的数据资源体系,支撑科研体系协同应用、科研领域创新应用等。

4 结论

本文研究了科研数据共享相关的制度机制、标准规范、数据模型、管理平台、应用架构等,通过综合运用数据资源无创采集、标准映射与集中管控,以及多源数据融合归一化维护管理、数据管理平台定制化组装等,有助于解决跨系统跨部门科研数据的统一采集、联动更新问题,实现数据源头一致、数据一致、更新一致,便于数据资源集成,有效管理数据资源、控制数据质量,提升应用效能。后续将在此基础上开展平台原型建设。

参考文献

[1] 杨帆.金融监管中的数据共享机制研究[J].金融监管研究,2019,94(10):53-68.
[2] 张贵香,刘桂锋,梁炜.我国科研数据管理理论与服务研究进展述评[J].情报理论与实践,2020(6):187-193.
[3] 司莉,邢文明.国外科学数据管理与共享政策调查及对我国的启示[J].情报资料工作,2013(1):61-66.
[4] 顾立平,陈新兰,张萧月,等.开放科研数据中的数据价值提升策略[J].图书馆论坛,2020,40(9):115-124.
[5] 邢文明,吴方枝,司莉.高校图书馆开展科研数据管理与共享服务调查分析[J].图书馆论坛,2013,33(6):19-25.
[6] 刘韫照,刘文江.世界一流高校科研数据管理政策研究[J].大学图书馆情报学刊,2020,38(3):31-41.
[7] 段雨泽,王其发.数据共享机制探讨[J].科学与信息化,2019(30):57.
[8] 李真.P2P 网贷信息用征信:金融分析与法律构建[J].当代经济管理,2015(7):85-91.
[9] 黄源,施栩婕,李晨英.基于科学数据管理流程的科研机构职责分析[J].数字图书馆论坛,2020,188(1):20-26.
[10] 金贞燕,孙华丽.科研数据管理相关权益分析[J].情报资料分析,2019,40(2):45-51.
[11] 祝风云.基于利益相关者视角的研究数据管理平台比较研究[J].图书馆学研究,2020(9):35-42.
[12] 司莉,李璐.我国高校科研数据共享中的知识产权与利益协调机制[J].图书馆,2018(7):18-24.
[13] 邹丹.铁路主数据管理平台关键技术研究[J].铁路计算机应用,2017,26(1):31-35.
[14] 司莉,曾粤亮.机构科研数据知识库联盟数据治理框架研究[J].图书馆论坛,2018(8):61-67.
[15] 王建民,王晨,刘英博,等.大数据系统软件创新平台与生态建设[J].大数据,2018(5):104-112.
[16] 陈君.主数据管理平台建设研究[J].铁道工程学报,2016,212(5):134-136.
[17] 甘彬,冯敏.基于 SOA 架构的主数据管理系统研究[J].自动化技术与应用,2020,39(3):42-44.
[18] 雷洁,赵瑞雪,李思经,等.知识图谱驱动的科研档案大数据管理系统构建研究[J].数字图书馆论坛,2020,189(2):19-27.
[19] 李萌,魏纬.基于 SOA 的主数据管理架构设计及实践[J].兵工自动化,2015,34(8):49-51.
[20] 谢婧,钱力,师洪波,等.科研学术大数据的精准服务架

(下转第 74 页)

(上接第 63 页)

metrics to determine the suitability and cost-effect of Self-* capabilities for autonomic computing systems[J]. IEEE Access,2019,7:139759-139772.
[12] SINGH S,CHANA I.QoS-aware autonomic resource management in cloud computing: a systematic review[J]. ACM Computing Surveys(CSUR),2015,48(3):1-46.
[13] BORTOLUSSI L,GALPIN V,HILLSTON J,et al.Hybrid semantics for PEPA[C]//2010 Seventh International Conference on the Quantitative Evaluation of Systems(QEST),Williamsburg,USA,15-18 Sept.,2010:181-190.
[14] HILLSTON J.Fluid flow approximation of PEPA models[C]// Proceedings of the 2nd International Conference on Quantitative Evaluation of Systems,Torino,IEEE Computer Society Press,September,2009:33-42.
[15] HUANG Y,KINTALA C,KOLETTIS N,et al.Software rejuvenation:analysis, module and applications[C]//Twenty-Fifth International Symposium on Fault-Tolerant Computing. Pasadena,CA,USA.27-30 Jun.,1995:381-390.

(收稿日期:2020-08-04)

作者简介:
孙日明(1983-),男,硕士,工程师,主要研究方向:计算机相关技术。

扫码下载电子文档

版权声明