

DOI:10.11918/j.issn.0367-6234.201903149

一种云计算中 Accrual 失效检测器

吴 晋¹,刘家希²,董 剑¹,左德承¹,赵 耀¹

(1. 容错与移动计算中心(哈尔滨工业大学),哈尔滨 150001; 2. 江苏省宽带无线通信和物联网重点实验室(南京邮电大学),南京 210003)

摘 要: 为更好地解决云计算中网络环境的动态性对失效检测性能的影响,提出一种适用于云计算环境的双滑动窗口 accrual 失效检测器(Two Windows Accrual Failure Detector, 2WA-FD). 首先,对不同网络环境下心跳消息到达时间间隔的概率分布进行实验分析,发现威布尔分布是一种更加合理的描述心跳消息到达时间间隔的概率分布模型,以此概率分布模型作为计算 accrual 失效检测器怀疑值的依据能够获得更高的精确度;其次,通过对 accrual 失效检测器实现架构的分析,通过采用双滑动窗口处理机制处理心跳消息能够更好地应对网络环境的突然变化;最后,通过开源实验数据以及租用云服务器搭建的实验平台比较和分析 5 种不同的 accrual 失效检测器实现的性能. 实验结果表明,该失效检测器较其他同类型失效检测器在同样的检测负载情况下,能够获得更好的检测速度与检测准确性. 因此,本文所提出的基于威布尔分布的双滑动窗口的 accrual 失效检测器 2WA-FD 能够准确、快速地发现云计算中的节点失效,有效地降低网络环境动态性对失效检测性能的影响.

关键词: 云计算;accrual 失效检测器;服务质量;威布尔分布

中图分类号: TP302.8 **文献标志码:** A **文章编号:** 0367-6234(2019)11-0016-06

An accrual failure detector in cloud computing

WU Jin¹, LIU Jiayi², DONG Jian¹, ZUO Decheng¹, ZHAO Yao¹

(1. Fault-tolerant and Mobile Computing Research Center (Harbin Institute of Technology), Harbin 150001, China;
2. Jiangsu Key Laboratory for Broadband Wireless Communication and Internet of Things(Nanjing University of Posts and Telecommunications), Nanjing 210003, China)

Abstract: In order to better solve the problem that the performance of failure detection is effected by dynamic of network environment in cloud computing, a new adaptive accrual failure detector (Two Windows Accrual Failure Detector, 2WA-FD) was proposed. First, two groups of actual data from two network conditions were analyzed, and we found that the Weibull distribution is a more reasonable distribution assumption for heartbeat inter-arrival time. According to the Weibull distribution, the suspicion level of accrual failure detector is more accurate. Second, the framework of accrual failure detector was analyzed and improved, and the suspicion level was calculated by two sliding windows. This framework is fit for dealing with the dynamic of network conditions. Finally, the 2WA-FD and other failure detectors were tested on open source experimental data and our experimental platform. The experimental results show that the 2WA-FD has better performance in terms of low detection time and high detection accuracy with the same detection overhead. Thus, the 2WA-FD can accurately and quickly find out the node failures in cloud computing, and effectively reduce the influence of dynamic on the performance of failure detection.

Keywords: cloud computing; accrual failure detector; quality of service; Weibull distribution

云计算通过组织成百上千台计算和存储服务器利用互联网为用户提供计算资源^[1-2]. 如此大的规模,结合不断增加的系统复杂性,使服务器的失效变得不可避免^[3]. 因此,为使系统能够在一些部件失效的情况下提供可靠和持续的服务,云计算采用了容错的设计架构^[4-5]. 失效检测器(Failure Detector, FD)是构成云计算容错架构的基础模块^[6]. 一个理

想的失效检测器应当同时提供可接受的服务质量(QoS)以及满足多个云计算应用的不同的 QoS 需求^[7].

由 defago^[8]提出的 accrual 检测器十分适合服务于大规模分布式系统. 它以连续怀疑值作为最终的输出结果,将监测权与解释权相分离. 在 accrual 检测器中,怀疑值的计算依靠历史心跳消息到达时间间隔的概率分布. 以往的实现中,通常假设心跳消息到达时间间隔的概率分布符合一些稳定的或变化缓慢的概率分布(如正态分布和指数分布^[9-10]),缺少对网络环境突然变化的考虑. 在云计算中,服务器

收稿日期: 2019-03-20
作者简介: 吴 晋(1986—),男,博士研究生;
董 剑(1978—),男,教授,博士生导师
通信作者: 董 剑,dan@hit.edu.cn

可能处于活跃、忙碌、过载、离线甚至崩溃状态^[11], 这些情况造成网络行为的突然改变. 并且, 通过对云计算中网络行为的分析, 发现其网络环境容易发生突然变化^[12].”

在本文中, 提出一种能够适应云计算环境中网络环境突然变化的 accrual 失效检测器—2WA-FD. 2WA-FD 使用两个滑动窗口去分别存储不同数量的心跳消息到达时间, 较大的滑动窗口处理网络环境的稳定期, 而较小的滑动窗口处理网络环境的突然改变. 同时, 通过对两种不同环境(WAN 和云计算环境)的实际数据的分析, 结果显示威布尔分布是一种对心跳消息到达时间间隔更加合适的分布假设. 因此, 使用威布尔分布对 2WA-FD 的历史心跳消息到达时间间隔作为分布概率假设.

为评价 2WA-FD 的性能, 选择了 5 个著名的失效检测器通过检测时间、平均错误发生概率以及查询准确率进行了比较. 所有参与比较的失效检测器均被实现在同样的 WAN 和云计算数据之上. 最终的实验结果显示, 2WA-FD 在不稳定的网络环境(特别是云计算)中, 能够拥有更加优秀的检测性能.

1 心跳消息行为分析

为找到一种更加合理的心跳消息到达时间间隔的概率分布假设, 选择两组不同平台(WAN 和云计算)的数据进行分析. 一组数据是曾经被 φ 检测器使用的来自 WAN 平台的经典的实验数据. 这组数据的采集时间超过一周, 并且有 500 多万条心跳消息. 其平均到达时间间隔为 103.9 ms, 标准差为 104.1 ms. 另一组数据来自租用亚马逊 EC2 平台产生的云计算实验数据. 这组数据的采集时间超过 3 个月, 并且有 300 万条心跳消息. 其平均到达时间间隔为 2.07 s, 标准差为 1.94 s.

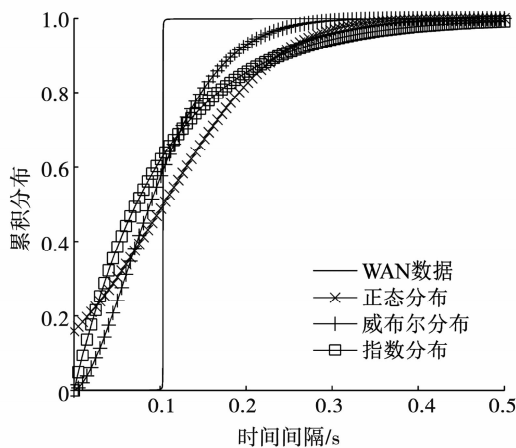


图 1 WAN 数据拟合结果

Fig. 1 Fitting result in WAN

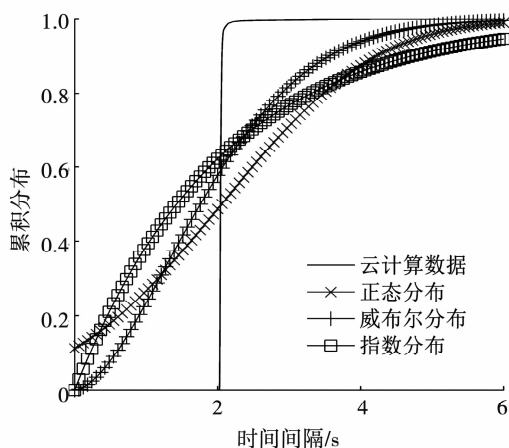


图 2 云计算数据拟合结果

Fig. 2 Fitting result in cloud computing

通过 MATLAB 中的 dfittool 工具对来自 WAN 平台的数据的累积分布进行了分析. 设定置信区间为 95%, 利用 3 种不同的分布(正态分布, 指数分布以及威布尔分布)进行数据拟合. 如图 1 所示, 曲线显示威布尔分布是最接近真实数据的分布. 类似的结果也出现在云计算平台的数据拟合后, 如图 2 所示. 根据实验结果, 可以得知威布尔分布是一种更加接近真实数据分布的概率分布. 因此, 在不稳定的网络环境中, 威布尔分布是一种更加合理的描述心跳到达时间间隔的概率分布.

2 2WA-FD 失效检测算法实现

2.1 系统模型

考虑一个部分同步的系统由有限个进程 $\Pi = \{p_1, p_2, \dots, p_n\}$ 组成. 每一个进程正常工作直到崩溃, 并且崩溃后不能恢复. 任意两个进程可以被一条不可靠的链路连接. 因为多数的失效检测器是用 UDP 协议实现的, 所以假设进程之间的连接链路是 fair-lossy 链路(即没有消息被复制或修改, 而且没有新的消息被创造). 如果一个进程 p 持续发送心跳消息 m 到 q , q 最终会收到心跳消息 m .

假设存在一些全局时间, 这些全局时间被定义为全局稳定时间(GST). 在到达全局稳定时间后, 消息的传输时间和进程的处理速度都是有界的, 但是这个界限及 GST 是未知的.

为简化描述, 考虑一个系统中仅包含两个进程 p 和 q , 进程 q 正在检测进程 p . 进程 p 每隔时间 Δt (发送间隔)发送一条心跳消息到进程 q . 如果进程 q 没有在一个时间周期内收到进程 p 的消息, 就会考虑进程 p 已经失效.

2.2 2WA-FD 检测器的实现

基于以上对心跳消息到达时间间隔的分析, 能

够假设心跳消息的到达时间间隔符合威布尔分布. 因此, 2WA-FD 的怀疑值 w_d 可通过以下公式.

$$w_d(T_{\text{now}}) = F(T_{\text{now}} - T_{\text{last}}). \quad (1)$$

式中 $F(t)$ 为威布尔分布的累积分布函数, 因此

$$F(t) = 1 - e^{-(t/\alpha)^\beta}. \quad (2)$$

式中 $t > 0$, 且参数 α 和 β 能够通过最小二乘法进行估算^[13].

为适应网络环境的突然变化, 用两个滑动窗口 (W_s 和 W_b) 去保存历史心跳消息到达的时间. 较小的滑动窗口 W_s 被用来存储少量的最近的心跳消息, 用来处理网络环境的突然改变 (可能由于服务器失效引起). 而较大的滑动窗口 W_b 能够存储更多的心跳消息的到达时间, 用来应对稳定的网络环境.

作为一个 accrual 失效检测器, 其实现方法相对简单. 预热期过后, 当新的心跳消息到达, 其到达时间被分别存储到不同的滑动窗口. 同时, 滑动窗口中最早的心跳消息到达时间被删除. 然后, 滑动窗口中的心跳消息的到达时间被用来计算威布尔分布的参数 α 和 β . 接下来, 基于式 (15) 和 (16), 能够分别计算当前的怀疑值 (w_s 和 w_b), 两者之间较小的值作为最后 2WA-FD 的输出值 w_d . 最后, 应用程序比较自己设定的阈值 W_d 与输出值 w_d , 它会根据比较结果执行相关的操作或者开始怀疑被检测进程. 详细的 2WA-FD 实现信息可以见图 5. 它是容易证明的, 2WA-FD 是一个最终完美失效检测器 $\diamond P_{ac}$.

在每次失效检测过程中, 如果检测器在规定时间内, 没有收到心跳消息, 怀疑值将会增长, 如算法 1 所示, Increased w_d , 其时间复杂度为 $O(1)$. 如果检测器接收到心跳消息并且心跳消息的序号大于目前滑动窗口中心跳消息的最大序号, 那么将会计算新的怀疑值. 在此过程中, 频度最高的语句是利用最小二乘法计算威布尔分布的参数 (Compute parameters α_s and β_s 以及 Compute parameters α_b and β_b), 其他语句的频度为常数. 计算威布尔分布的参数时间复杂度分别为 $O(W_s)$ 和 $O(W_b)$. 那么, 根据时间复杂度的求和法则, 可知 2WA-FD 执行一次失效检测的时间复杂度为 $O(W_b)$. 综上所述, 在 n 次失效检测过程中, 2WA-FD 算法的时间复杂度为 $O(W_b \cdot n)$. 同时, 由于 2WA-FD 中滑动窗口值为常数 (通常 W_b 设定为常数值, 本文实验中设定为 10^3 和 10^4), 也可认为 2WA-FD 的算法复杂度为 $O(n)$.

算法 1 2WA-FD 检测器算法

Initialization:

Δt : sending interval

$sn_1 \leftarrow 0$; /* keep the largest sequence number */

$W_s = W_b = \perp$; /* empty sliding windows for inter-

arrival times */

$T_{\text{last}} \leftarrow 0$; /* last arrival time of heartbeat */

Process p :

For all $i > 0$, at time $(i \cdot \Delta t)$: send heartbeat m_i to q ;

Process q :

Task 1: if q did not receive heartbeat during a certain time

Period of q 's clock

Increase w_d ; /* suspect p */

Task 2: upon receiving heartbeat m_j from p

If $j > sn_1$ **then**

$W_s \leftarrow (T_{\text{now}} - T_{\text{last}})$;

$W_b \leftarrow (T_{\text{now}} - T_{\text{last}})$;

Compute parameters α_s and β_s ;

Compute parameters α_b and β_b ;

$w_s \leftarrow 1 - e^{-((T_{\text{now}} - T_{\text{last}})/\alpha_s)^{\beta_s}}$;

$w_b \leftarrow 1 - e^{-((T_{\text{now}} - T_{\text{last}})/\alpha_b)^{\beta_b}}$;

$w_d \leftarrow \min(w_s, w_b)$;

$T_{\text{last}} \leftarrow T_{\text{current}}$;

$sn_1 \leftarrow j$;

Application k :

Compare w_d with the threshold W_d^k (from application requirement);

Carry out some actions or start to suspect p ;

3 实验验证及分析

对采用不同尺寸滑动窗口的 2WA-FD 性能进行了评估, 并且找出 W_s 和 W_b 理想的参数配置. 其次, 把 2WA-FD 和自适应失效检测器进行比较分析. 所有实验均执行在来自 WAN 平台和云计算实验平台的传输日志文件之上. 实验参考了文献[11]的方法, 利用传输日志文件对各个失效检测器进行重演, 这样能够保证所有失效检测器的实现在同样的网络环境下.

在 WAN 环境中, 实验包括两台计算机: 一台位于日本, 而另一台位于瑞士. 两台计算机通过正常的洲际互联网连接进行通信. 一台计算机负责周期性地发送心跳消息, 而另一台计算机负责记录每一条心跳消息到达的时间. 在实验过程中没有计算机发生失效. 心跳消息的实际发送频率是每 103.501 ms 发送一条心跳消息. 在实验中, 心跳消息的往返时间以较低的概率进行测量, 平均 RTT 时间是 283.3 ms (标准差为 207.3 ms, 最小值为 270.2 ms 以及最大值为 717.8 ms). 超过 500 万条心跳消息被接收, 且丢失率约为 0.4%.

在云计算环境中, 通过租用亚马逊 EC2 的服务

器搭建实验平台. 云端服务器分别位于东京、新加坡以及俄勒冈(美国)的 3 个节点上. 3 个服务器节点都可以对用户提供一个相同的基于 Web 的查询服务, 为便于测试, 并没有设置前端处理机, 假设用户客户端已经知道 3 个服务器的网络地址, 当前服务节点失效后, 会自动连接新的服务器. 这些服务器配有 2.5 GHz 的 Intel Xeon 的处理器, 1 GHz 内存以及搭载 Red Hat Linux 7.2 操作系统. 实验环境如图 3 所示. 在实验中, 位于哈尔滨的客户端首先连接位于俄勒冈的服务器获取服务. 通过故障注入的方法使服务器停止服务(同时终止检测模块的运行), 客户端依次访问新加坡和东京的服务器. 实验进行近 3 个月, 发送间隔被设定为 2 s, 而实际测量的发送频率为 2.092 s(标准差为 0.019 s, 最小值为 1.964 s 以及最大值为 5.239 s). 在实验中, 平均 RTT 时间为 0.179 2 s, 标准差为 0.008 6 s, 最小值为 0.118 7 s 以及最大值为 14.505 s. 超过 300 万条心跳消息被接收, 丢失率大约为 0.72%.

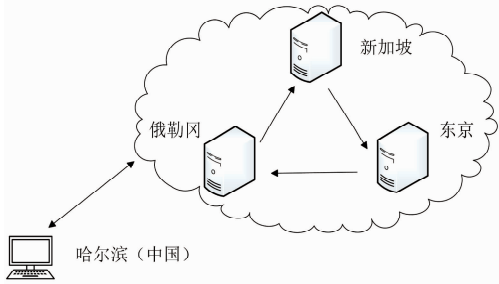


图 3 云计算实验环境

Fig. 3 Experiment of cloud computing

在实验中, 检测时间, 平均错误发生概率以及查询准确率作为主要的失效检测器的性能指标. 对于 2WA-FD, 其阈值设定范围为 $W_d \in (0, 1)$ (如 $W_d = 0.01$ 代表设定的检测器的平均错误发生概率为 10^{-2}). 根据相应的阈值设定检测器参数, 在每次实验中能获取不同的检测时间, 平均错误发生概率以及查询准确率.

在所有的实验中, 可以通过估计来计算检测时间的均值^[9]. 假设一个进程刚好在发送完一条心跳消息后崩溃(最糟糕的情况), 可以测量此刻到检测器发出怀疑的时间. 对于文中所提到的失效检测器, 根据阈值的设定去反向求取心跳消息的超时值 Δ_{to} . 此外, 根据心跳消息的平均 RTT 时间, 可以得知心跳消息单程传输的时间 Δ_{tr} . 那么, 就可以计算平均检测时间的值^[9,11]

$$T_D \approx \Delta_{tr} + \Delta_{to}. \quad (3)$$

3.1 滑动窗口尺寸影响

此实验检测滑动窗口尺寸对 2WA-FD 检测性

能的影响. 通过改变滑动窗口的尺寸从而获取不同的准确性. 为了更加清楚的观察实验结果, 仅截取了最明显的部分进行描述.

图 4 中显示了 WAN 环境下平均错误发生概率与检测时间的关系. X 坐标轴表示检测时间, Y 坐标轴表示平均错误发生概率. 图 5 中显示了 WAN 环境下检测时间与查询准确率的关系. 当 $W_b = 10^4$, $W_s = 10$ 时, 2WA-FD 能够获得最低的平均错误发生概率. 当 $W_b = 10^4$, $W_s = 10$ 和 $W_b = 10^3$, $W_s = 10$ 时, 2WA-FD 能够获得最高的查询准确率.

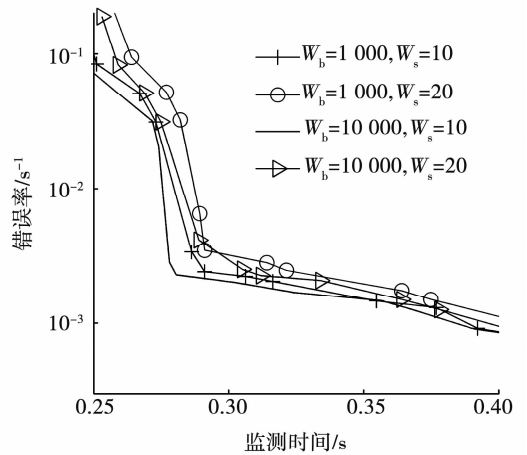


图 4 WAN 环境中不同尺寸滑动窗口下平均错误发生概率与检测时间

Fig. 4 Mistake rate vs. detection time of different sliding windows in WAN

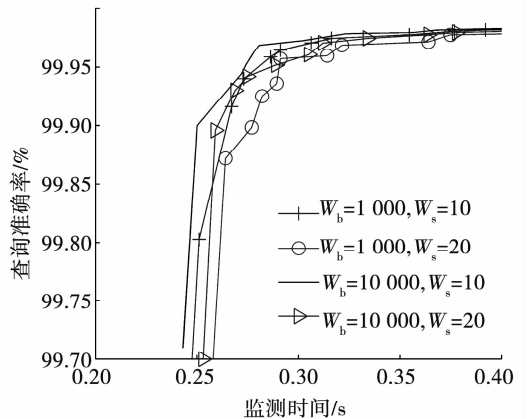


图 5 WAN 环境中不同尺寸滑动窗口下查询准确率与检测时间

Fig. 5 Query accuracy vs. detection time of different sliding windows in WAN

图 6 和 7 显示了云计算中的实验结果. 从实验结果可知, 当 $W_b = 10^4$, $W_s = 10$ 时, 2WA-FD 能够获得最好的性能.

以上的实验证明 2WA-FD 的性能会受到滑动窗口尺寸的影响. 根据平均错误发生概率和查询准确率, 本文所提出的失效检测器随着 W_b 的增长和

W_s 的降低而表现的更好. 同时, 在实验结果中, 同时注意到, 当 W_b 的尺寸相同时, 更小的 W_s 带来更好检测性能. 而当 W_s 固定时, 随着 W_b 的增长, 检测器性能改善不明显. 考虑与其他检测器的性能比较, 对于 2WA-FD, 选用 $W_b = 10^3$, $W_s = 10$ 配置进行实验.

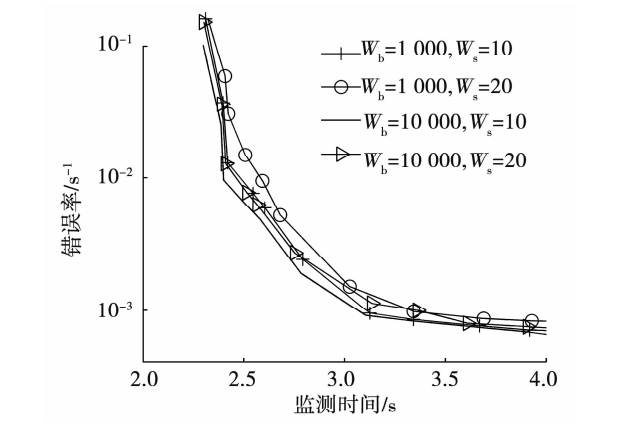


图 6 云计算环境中不同尺寸滑动窗口下平均错误发生概率与检测时间

Fig. 6 Mistake rate vs. detection time of different sliding windows in cloud computing

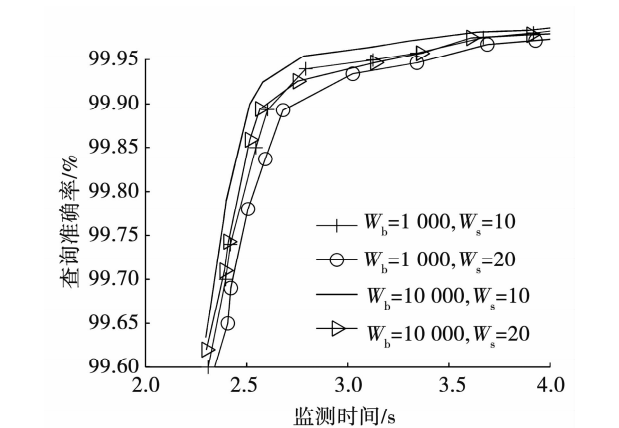


图 7 云计算环境中不同尺寸滑动窗口下查询准确率与检测时间

Fig. 7 Query accuracy vs. detection time of different sliding windows in cloud computing

3.2 不同的失效检测器的比较

在此实验中, 选用 5 个著名的失效检测器与 2WA-FD 进行比较, 并且选择了不同的检测参数来获得每一个失效检测器的性能. 不同于其他失效检测器, Bertier 失效检测器没有调整参数, 因此它的性能在图中只有一个点表示. 各个失效检测器的配置参数如下: 对于 Chen 失效检测器, 参数设置如文章^[9]中, $\alpha \in [0, 10^3]$; 对于 SFD, 其参数设置如文章^[11]中, $SM_1 = \alpha$; 对于 ϕ 失效检测器, 其参数设置如文章^[9]中, $\phi \in [0.5, 16]$; 对于 ED FD, 其参数设置如文章^[10]为 $E_d \in [10^{-4}, 10]$. 这 4 种失效检测器的滑动窗口都设置为 10^3 .

图 8 显示了 WAN 环境下的平均错误发生概率与检测时间的关系. X 坐标轴代表检测时间, Y 坐标轴代表平均错误发生概率. 图 9 显示了 WAN 场景下查询准确率与检测时间的关系.

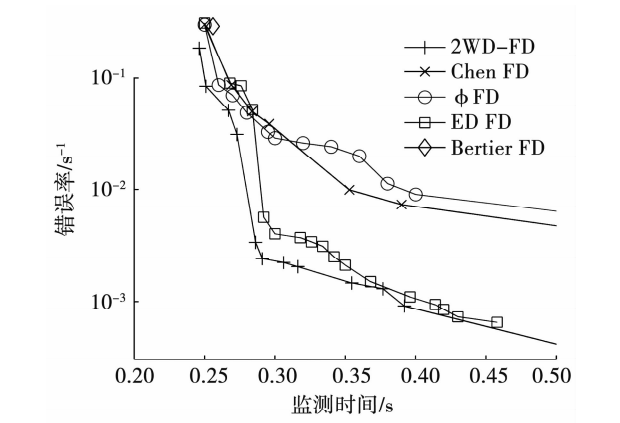


图 8 WAN 环境下平均错误发生概率与检测时间的关系

Fig. 8 Mistake rate vs. detection time in WAN

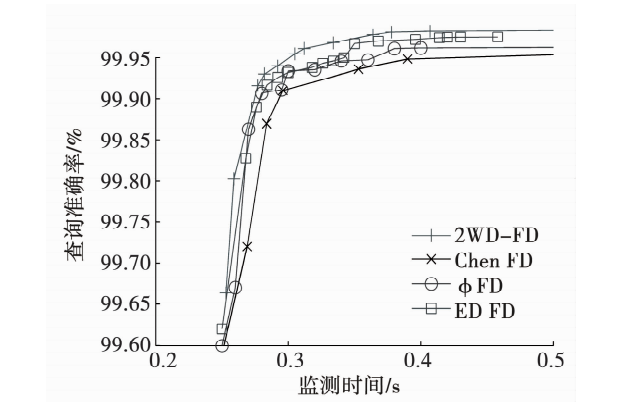


图 9 WAN 环境下查询准确率与检测时间的关系

Fig. 9 Query accuracy vs. detection time in WAN

实验结果显示所有的失效检测器都遵循相同的趋势. 随着检测时间的增长, 平均错误发生概率呈现下降的趋势, 而查询准确率呈现上涨的趋势. 但是, 在 WAN 环境中, 当 $T_D < 0.4$ s 时, 2WA-FD 的性能优于其他失效检测器. 在平均错误发生概率方面, 2WA-FD 相对于 ED FD 有最多 40% 的提升. 而且, 在大多数检测时间上, 2WA-FD 获得了最好的查询准确率.

图 10 和 11 显示了在云计算环境中平均错误发生概率, 查询准确率与检测时间的关系. 类似于 WAN 环境中的结果, 2WA-FD 在云计算环境中获得了最低的平均错误发生概率和最高的查询准确率. 当 $2.5 \text{ s} < T_D < 4 \text{ s}$ 时, 2WA-FD 的平均错误发生概率明显低于其他失效检测器 (有大约 45% 的提升). 在云计算环境中, 2WA-FD 受益于双滑动窗口, 在不稳定的网络环境中获得了良好的性能.

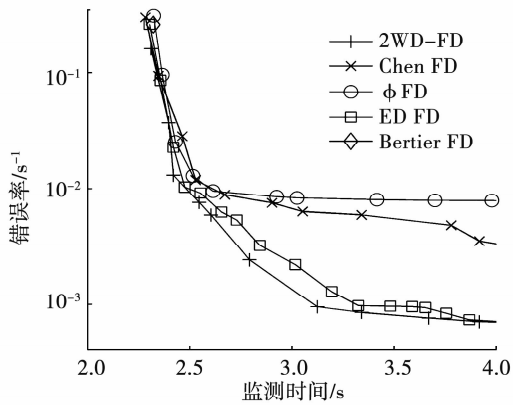


图 10 云计算环境下平均错误发生概率与检测时间的关系

Fig. 10 Mistake rate vs. detection time in cloud computing

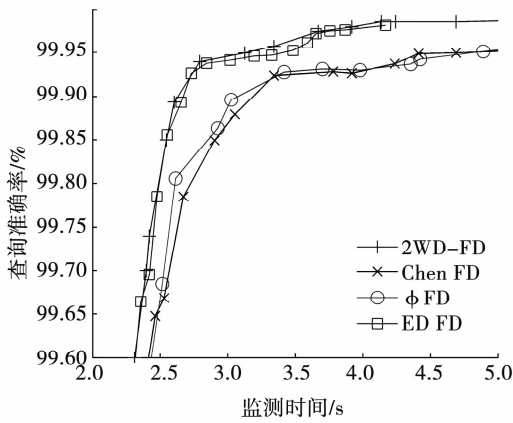


图 11 云计算环境下查询准确率与检测时间的关系

Fig. 11 Query accuracy vs. detection time in cloud computing

从实验中,可以推断出 2WA-FD 在不稳定和动态的网络环境(例如 WAN 和云计算)中相较于其他失效检器获得了最好的检测性能. 在 2WA-FD 中,两个滑动窗口能更好的适应网络环境的变化. 但是,更大的 W_b 并不能明显的提升 2WA-FD 的性能(如图 5,6,7 和 8 所示). 滑动窗口尺寸的大小应该取决于用户的特殊需求与配置.

4 结 论

失效检测器在可靠的分布式系统中扮演了非常重要的角色. 在本文中,提出了双窗口的 accrual 失效检测器 2WA-FD. 通过采用双滑动窗口存储心跳消息到达时间,以及更加合理的心跳到达时间间隔的分布假设,2WA-FD 能够更好的应对网络环境的突然变化,并且能够很好的适应云计算的网络环境. 通过对比实验,结果显示本文所提出的失效检测器在检测时间,平均错误发生概率以及查询准确率方面表现出更好的性能. 因此,2WA-FD 适合于部署在云计算中提供失效检测服务.

参考文献

[1]PANNU H S, LIU Jianguo, GUAN Qiang, et al. AFD: Adaptive failure detection system for cloud computing infrastructures [C]// Proc 31st IEEE International Performance Computing and Communications Conference. Austin, Texas, USA: IEEE Press, 2012: 71. DOI:10.1109/PCCC.2012.6407740

[2]DABBAGH M, HAMDAROU B, GUIZANI M, et al. Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment[J]. IEEE network, 2015, 29(2): 56. DOI:10.1109/MNET.2015.7064904

[3]ZHOU Ao, WANG Shangguang, ZHEN Zibin, et al. On cloud service reliability enhancement with optimal resource usage [J]. IEEE Transactions on Cloud Computing, 2016, 4(4): 452. DOI: 10.1109/TCC.2014.2369421

[4]FETZER C, RAYNAL M, TRONEL F. An adaptive failure detection protocol [C]// Proc the 2001 Pacific Rim International Symposium on Dependable Computing. Seoul, Korea: IEEE Press, 2001: 146. DOI:10.1109/PRDC.2001.992691

[5]DING Yongsheng, YAO Guang shun, HAO Kuangrong. Fault-tolerant elastic scheduling algorithm for workflow in Cloud systems [J]. Information Sciences, 2017, 393: 47. DOI:10.1016/j.ins.2017.01.035

[6]LIN Rongsheng, WU Budan, YANG Fangchun, et al. An efficient adaptive failure detection mechanism for cloud platform based on volterra series[J]. China Communications, 2014, 4(11): 1. DOI: 10.1109/CC.2014.6827564

[7]LAVINIA A, DOBRE C, Pop F, et al. A failure detection system for large scale distributed systems [C]// Proc the 2010 International Conference on Complex, Intelligent and Software Intensive Systems. Krakow, Poland: IEEE Press, 2010: 482. DOI:10.1109/CISIS.2010.29

[8]DEFAGO X, URBAN P, HAYASHIBARA N, et al. Definition and specification of accrual failure detectors [C]// Proc the International Conference on Dependable Systems and Networks. Yokohama, Japan: IEEE Press, 2005: 206. DOI:10.1109/DSN.2005.37

[9]HAYASHIBARA N, DEFAGO X, YARED R, et al. The φ accrual failure detector [C]// Proc the 23rd IEEE International Symposium on Reliable Distributed Systems. Florianopolis, Brazil: IEEE Press, 2004: 66. DOI:10.1109/RELDIS.2004.1353004

[10]TOMSIC A, SENS P, GARCIA J, et al. 2W-FD: a failure detector algorithm with QoS [C]//Proc the 29th International Parallel and Distributed Processing Symposium. Hyderabad, India: IEEE Press, 2015: 885 - 893. DOI: 10.1109/IPDPS.2015.7410.1109/IPDPS.2015.74

[11]XIONG Naixue, VASILAKOS A V, WU Jie, et al. A self-tuning failure detection scheme for cloud computing service [C]// Proc the 26th International Parallel and Distributed Processing Symposium. Shanghai, China: IEEE Press, 2012: 668. DOI:10.1109/IPDPS.2012.126

[12]DALMAZO B L, VILELA J P, CURADO M. Online traffic prediction in the cloud [J]. International Journal of Network Management, 2016, 26(4): 269. DOI:10.1002/nem.1934

[13]BHATTACHARYA P, BHATTACHARJEE R. A study on Weibull distribution for estimating the parameters [J]. Journal of Applied Quantitative Methods, 2010, 5(2): 234. DOI: 10.1109/TR.1981.52265