

DOI: 10.11918/j.issn.0367-6234.201708010

一种车辆云计算中共享失效检测器

刘家希,董 剑,吴智博,吴 晋,温东新,赵 耀

(哈尔滨工业大学 容错与移动计算中心, 哈尔滨 150001)

摘要:为更好地解决车辆云计算中车辆的高度动态性对失效检测性能的影响,提出了一种适用于车辆云计算环境的基于检测结果共享机制的失效检测器.首先,对车辆云计算的系统架构进行了分析,发现车辆云计算系统具有明显的层次结构,而且可以利用路侧单元对车辆节点进行分组.在此基础上,引进检测结果共享机制,提出针对车辆云计算环境的共享失效检测器 VC-FD.然后,构建了可以对所提出的失效检测器进行定量分析的共享失效检测模型,该模型主要针对检测时间、平均错误发生概率以及检测负载三个失效检测服务质量指标进行评价.最后,在仿真环境下比较和分析了所提出的共享失效检测器与非共享失效检测器的性能.实验结果表明,该失效检测器在保证检测准确性的前提下,通过增加有限的检测负载能够明显地改善系统中节点失效检测时间.因此,提出的基于检测结果共享机制的失效检测器 VC-FD 能够准确、快速的发现车辆云计算中的节点失效,有效地降低车辆高度动态性对失效检测性能的影响.

关键词: 车辆云计算; 系统架构; 路侧单元; 服务质量; 共享失效检测

中图分类号: TP302.8

文献标志码: A

文章编号: 0367-6234(2018)05-0024-06

A cooperative failure detector in vehicular cloud computing

LIU Jiayi, DONG Jian, WU Zhibo, WU Jin, WEN Dongxin, ZHAO Yao

(Harbin Institute of Technology, Fault-tolerant and Mobile Computing Research Center, Harbin 150001, China)

Abstract: To solve the problem that the performance of failure detection is effected by mobility of vehicular cloud computing, a new failure detector (VC-FD) based on architecture of vehicular cloud computing was proposed. We found that vehicular cloud computing has a hierarchical structure and the vehicles can be grouped through Roadside Unit (RSU), according to the architecture analysis of vehicular cloud computing. On this basis, the sharing mechanism of detection results is introduced, and the new failure detector VC-FD for vehicular cloud computing was proposed. Then, the model of cooperative failure detection was proposed to evaluate the detection time, mistake rate and detection overhead. The performance of the VC-FD and non shared failure detector was compared in a simulation environment. The experimental results show that the VC-FD can ensure the accuracy, and significantly improve the node failure detection time by increasing limited detection overhead. Thus, the VC-FD can accurately and quickly find out the node failures in vehicular cloud computing, and effectively reduce the influence of vehicular mobility on the performance of failure detection.

Keywords: vehicular cloud computing; architecture; RSU; quality of service; cooperative failure detection

目前,随着移动云计算(Mobile Cloud Computing, MCC)以及车联网(Vehicular Networking, VN)的发展,出现一种融合这两种技术的新的通信、计算平台—车辆云计算(Vehicular Cloud Computing, VCC)^[1].在车辆云计算中,车辆既是服务的提供者,也是服务的消费者^[2-3].车辆云计算一个重要的特征就是高度的动态性^[3-4],车辆的突然加入或者离开都会造成正在进行的任务突然中断^[5].我们需要解决这种动态性并且提供一种高效的控制架构用以引导服务状态以及云资源^[6-7].大量的自适应失

效检测器^[8-9]作为实现这种控制架构的基础构件被提出.为进一步提高失效检测性能,在系统层面提出了基于检测结果共享机制的失效检测(简称共享失效检测).共享失效检测^[10-12]是不同节点之间通过检测结果共享机制,改善失效检测性能.基于层次式的检测方法^[10-11]和基于Gossip式的检测方法^[12]是共享失效检测的两种重要的方法.但这些共享失效检测算法并没有考虑车辆云计算的系统架构特点.在保证失效检测准确性的前提下,为进一步提高检测速度应对车辆云系统的高度动态性,即快速发现已经离开系统的车辆节点,根据车辆云计算的系统架构特点,提出共享失效检测器 VC-FD (VCC Cooperative Failure Detector).通过仿真实验,比较了 VC-FD 和非共享失效检测器的性能.

收稿日期: 2017-08-03

基金项目: 国家自然科学基金(61100029 和 61370085)

作者简介: 刘家希(1988—),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

1 系统模型

1.1 车辆云计算系统架构

Olariu 和 Arif 等^[13-14]提出车辆云计算的概念,车辆云计算结合计算、传感器、通信与存储资源于一体向授权用户提供服务.在车辆云计算中,车辆与云服务数据中心通过 V2I (Vehicles - to - Infrastructure) 机制进行通信(例如 LTE, WiMax 等).而且,车辆也能够以独立方式工作,仅需依靠 V2V (Vehicles-to-Vehicles) 机制进行通信.为改善连接性,路侧单元(Roadside Unit, RSU)被部署于车辆网络边缘^[15].因此,车辆云计算可被认为是由松散的二层架构组成.用户可以从一级数据中心或者二级车辆朵云(Cloudlet)获取云服务.车辆朵云是由数台通过 V2I 或者 V2V 通信连接的车辆和 RSU 所组成.如图 1 所示是一种车辆云计算的系统架构^[2,7].

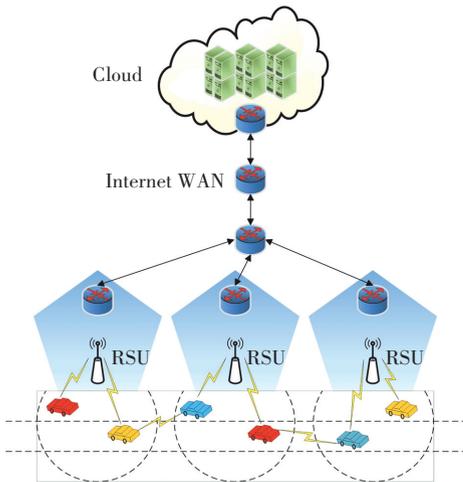


图 1 车辆云系统架构

Fig.1 Architecture of vehicular cloud computing

1.2 理论模型

考虑一个部分同步的系统由有限个节点 $\Pi = \{p_1, p_2, \dots, p_n\}$ 组成.每一个节点正常工作直到崩溃(可认为车辆所提供资源不可用),并且崩溃后不能恢复.任意两个节点可以被一条不可靠的链路连接.因为多数的失效检测检测器是用 UDP 协议实现的,所以我们假设进程之间的连接链路是 fair-lossy 链路^[16](即没有消息被复制或修改,而且没有新的消息被创造).如果一个节点 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(Global Stabilization Time, GST).在到达全局稳定时间后,消息的传输时间和节点的处理速度都是有界的,但是这个界限及 GST 是未知的.

1.3 失效检测器服务质量(QoS)指标

许多实际应用对失效检测器所提供节点的怀疑信息有时间的限制.它们无法接受过慢的或者太多错误的失效检测服务.为解决这个问题,Chen 提出了一系列描述失效检测器 QoS 的指标^[17].其中包括:检测到真实节点失效的速度以及避免错误检测的频率.

检测时间(Detection time, T_D):代表一个节点从崩溃到被永久怀疑的时间.

平均错误发生概率(Mistake rate, λ_M):代表失效检测器发生错误的概率.也就是正常工作的节点被失效检测器错误的怀疑.

检测负载(Detection overhead, O_D):代表为检测失效节点而产生的流量.对于一个监测节点检测失效节点所耗费的负载,可以用单位时间内产生的平均消息数量来衡量检测负载.

2 共享失效检测算法实现

2.1 基础检测方法

为描述方便,考虑系统由两个节点组成,节点 q 检测节点 p .节点 q 以间隔时间 τ 为周期向被检测节点 p 发送心跳消息“are you alive”,节点 p 收到以后回复应答消息“I am alive”以表明其处于工作状态.如果节点 q 在时间 τ 内没有收到应答消息,则考虑心跳消息丢失.那么在此之后,节点 q 至多连续发送 $k-1$ 个心跳消息,如果均没有收到应答消息,则推断节点 p 处于失效状态.否则,推断节点 p 处于正常状态.这种重传机制可以有效地减少心跳消息丢失对检测结果输出的影响,提高失效检测的准确性.检测参数 k 和 τ 可根据需要改变,决定失效检测器的性能.

2.2 共享失效检测器 VC-FD 原理

共享失效检测器 VC-FD 的目的是让目标节点的监测节点作为一个组合彼此协作,以便每个节点能够更快地获取目标节点的状态信息.在车辆云计算中,由于其系统架构的特点,可利用朵云对节点进行分组.每个朵云为一个分组,分组内的节点互相检测,如果发现节点失效,会共享失效信息到组内其他节点以及 RSU.而不同分组之间可通过 RSU 实现跨组交换信息,节点可通过自己的 RSU 查询其他分组的节点的状态.对于 RSU,它的状态由云端负责检测,如果 RSU 发生失效,分组内的节点会就近划分到附近的分组内,云端会将失效信息进行广播,以便系统中其他的 RSU 获知(在实际应用中,RSU 的可靠性远远高于车辆节点的可靠性).

图 2 显示了共享失效检测器 VC-FD 的工作原

理. 对于同一分组内节点的失效检测情况, 既可根据自己的检测结果进行判断, 也可根据其他节点共享的检测结果进行判断. 例如, 节点 A1、A2、A3 及 A4 为同一分组的节点, 现 A1、A2 及 A4 都对 A3 进行检测, 恰好 A1 经过检测已经知道了 A3 失效, 那么它将会立即把失效信息通知节点 A2、A4 以及本组的 RSU. 对于不同分组间的节点失效检测情况, 可以通过 RSU 跨组交换信息得到目标节点的状态信息. 例如, 节点 A1 想要获取节点 C3 的状态, 它不会直接对 C3 进行检测, 而是通过本组的 RSU 去访问 C3 所在分组的 RSU, 从而获得 C3 的状态. 对于 RSU 的失效检测情况, 云端会检测 RSU 的状态, 并且广播 RSU 的失效信息到系统中其他的 RSU. 例如, 节点 B1 所在分组的 RSU 发生失效, 云端会广播失效信息到系统中其他 RSU, B1 同分组内的节点 B2、B3 及 B4 会根据距离远近就近分配到附近的分组中. 由于云计算中心的高可靠性, 不考虑云端的失效情况.

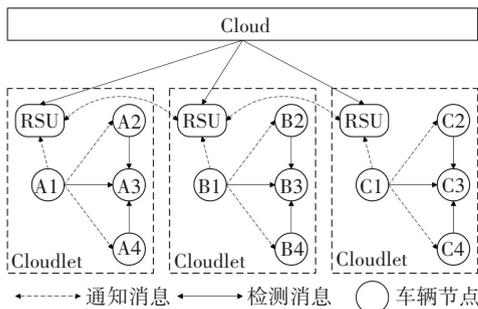


图 2 共享失效检测架构

Fig.2 Architecture of cooperative failure detection

2.3 共享失效检测理论模型

为更好地理解共享失效检测器 VC-FD, 给出了正式的理论分析. 方便定量的描述失效检测的 3 个 QoS 指标: 检测时间 T_D , 平均错误发生概率 λ_M 以及检测负载 O_D .

2.3.1 检测时间

对于基础的(非共享)检测方法, 如果心跳消息数量的丢失超过阈值 k , 监测节点 q 就可判断目标节点 p 发生失效. 考虑两种极端的情况, 即最短的检测时间和最长的检测时间. 如果节点 p 在收到节点 q 的心跳消息后立即失效, 那么, 节点 q 的检测时间需要 $(k-1)\tau$; 如果节点 p 在发送完应答消息后立即失效, 那么, 节点 q 的检测时间需要 $k\tau$. 因此, 节点 q 的失效检测时间在 $(k-1)\tau$ 和 $k\tau$ 之间(见图 3). 假设节点发生失效概率在 $(0, \tau)$ 上符合均匀分布, 那么非共享检测方法的平均节点失效检测时间为

$$T_D = (k - \frac{1}{2})\tau. \quad (1)$$

对于共享的检测方法, 如果心跳消息数量的丢失超过阈值 k , 或者监测节点收到其他邻居节点的通知消息, 监测节点 q 就可以判断目标节点 p 发生失效. 考虑两种极端的情况, 即最长的检测时间和最短的检测时间. 如果节点 q 在检测过程中没有收到其他邻居节点的通知消息, 那么节点 q 的最长检测时间为 $k\tau$. 如果节点 q 在即将发送心跳消息到目标节点时, 邻居发送通知消息, 那么节点 q 的最短检测时间近乎为 0. 实际上, 对 d 个邻居中至少一个邻居的通知消息到达节点 q 的平均时间更感兴趣. 因为在 $(0, k\tau)$ 上, 邻居节点通知消息在平均时间到达的概率远高于 0 时刻到达的概率. 根据次序统计理论^[18], 在 $(0, k\tau)$ 上均匀分布的 d 个邻居节点中至少一个邻居节点发送通知消息到节点 q 符合 $k\tau\beta_d$ 分布, β_d 是拥有参数 1 和 d 的贝塔分布, 其均值为 $1/(d+1)$. 因此, 节点 q 收到来自邻居节点之一的通知消息的平均时间为 $k\tau/(d+1)$. 考虑到分组内没有邻居节点的情况, 假设节点发生失效概率在 $(0, \tau)$ 上符合均匀分布, 那么共享检测方法的平均节点失效检测时间为

$$T_D = \min((k - \frac{1}{2})\tau, \frac{d+3}{2(d+1)}k\tau). \quad (2)$$

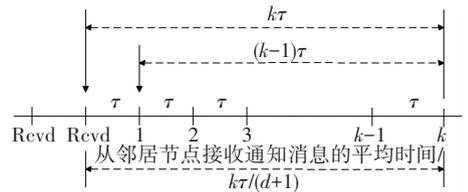


图 3 检测时间

Fig.3 Detection time

共享检测方法的失效检测时间既受阈值 k 和心跳发送间隔 τ 的影响, 又受邻居节点数量 d 的影响. 当邻居节点数量越多, 失效检测时间越短.

2.3.2 平均错误发生概率

对于基础的(非共享)检测方法, 发生错误检测是由于被发送到目标节点的连续 k 个心跳消息丢失, 因此, 其平均错误发生概率为

$$\lambda_M = 2p_1^k. \quad (3)$$

式中 p_1 代表心跳消息丢失的概率.

对于共享的检测方法, 发生错误检测是由于被发送到目标节点的连续 k 个心跳消息丢失或者收到邻居节点错误的通知消息. 那么, 共享检测方法的平均错误发生概率为

$$\lambda_M = 2p_1^k + 1 - (1 - 2p_1^k)^d. \quad (4)$$

从等式(3)和(4)可知, 共享检测方法的平均错误发生概率要高于非共享检测方法. 但是, 随着阈值 k 和邻居节点数量 d 的增加, 两种方法的平均错误发

生概率趋于一致.

2.3.3 检测负载

对于基础的(非共享)检测方法,如果目标节点没有失效,一个监测节点的负载是每个发送周期 τ 产生两条心跳消息,那么非共享方法的节点检测负载大约是

$$O_D = \frac{2(1 - f_p)}{\tau}. \quad (5)$$

式中 f_p ($0 < f_p < 1$) 是节点单独失效的概率.

对于共享的检测方法,增加的负载来自通知消息. 当目标节点失效(概率为 f_p), 监测节点将发送 $d - 1$ 条通知消息到其他邻居节点. 当目标节点工作正常(概率为 $1 - f_p$), 由于消息丢失(概率为 $2p_1^k$) 引起错误检测, 监测节点将发送 $d - 1$ 条通知消息到其他邻居节点. 那么共享检测方法的节点检测负载为

$$O_D = \frac{2 + df_p - 2f_p}{\tau} + \frac{(1 - f_p)(d - 1)}{\tau} 2p_1^k. \quad (6)$$

2.4 VC-FD 算法实现

基于上述思想, VC-FD 共享失效检测算法可由算法 1 中的伪代码表示. 为便于描述算法 1, 从监测节点 q 和被检测节点 p 的角度进行描述. 在实际系统中, 监测节点也同时是被检测节点, 所以, 算法 1 中的两个模块同时运行在每一个系统节点之上. 在算法 1 初始时, 确定心跳发送间隔 τ 以及心跳重传次数的阈值 k . 对于监测节点 q 而言, 如果收到其他邻居节点对节点 p 的状态的通知消息, 会推断 p 处于失效状态. 否则在每个心跳发送间隔发送检测消息 m_q 到 p . 如果发送 k 个检测消息 m_q 后, 没有收到任何回复, 则判定节点 p 失效, 并发布通知消息到邻居节点. 对于被检测节点 p 而言, 每当收到来自节点 q 的检测消息 m_q 后, 发送应答消息 m_a 到 q 以表明自己的状态.

算法 1 VC-FD 共享失效检测算法

Initialization:

τ : sending interval;

k : threshold;

1: for node q : (Monitoring node)

2: if receive notification message

3: $p \in \text{suspectlist}$;

4: for all $i \geq 1$, at time $(i \cdot \tau)$

5: send heartbeat message m_q to p (at most k);

6: if don't receive any acknowledge message

7: $p \in \text{suspectlist}$ and

send notification message to neighbor nodes;

8: for node p : (Detected node)

9: upon receive m_q from q do

10: send m_a to q .

3 实验验证及分析

为对提出的共享失效检测器 VC-FD 的性能进行验证, 基于网络仿真工具 NS2 设计共享失效检测算法的实验方案. 在实验中, 利用网络拓扑生成器 GT-ITM 生成 2 000 个节点的 transit-stub 模型作为底层网络. 车辆节点被随机分布在 stub 区域内. 传输延迟参考在 Internet 上所得到的数据, 两个节点之间的传输延迟是它们之间最短路径上的链路延迟之和. 它的范围从 1 ms ~ 220 ms, 平均值等于 96 ms. 任意一个节点通过自身的路由表可以访问到 d 个其他的节点, 这些节点称为其邻居节点, d 的取值范围在 $[2, 30]$ 之间均匀分布. 实验开始时由 200 个节点组成. 然后其他节点动态地加入和离开, 节点加入和离开遵循泊松分布 ($\lambda = 0.2/\text{s}$). 每次实验持续 3 个小时. 用 Gilbert 模型模拟网络连接的包丢失, 平均链路丢包率为 $p_1 = 1\%$. 如果点到点之间的路径由 m 个链接组成, 那么其丢包率为 $p = 1 - (1 - p_1)^m$. 实验中, 心跳发送间隔 τ 被分别设定为 0.5 s 和 1 s.

在这个实验环境下, 我们对共享失效检测算法的检测速度、检测准确性以及检测负载进行了验证和对比分析. 对比实验参照对象即为非共享失效检测方法.

3.1 检测时间

图 4 显示在不同的心跳发送间隔 ($\tau = 0.5$ s 和 $\tau = 1$ s) 情况下, 共享方法和非共享方法平均检测时间的比较. X 轴代表阈值 k (即节点间检测的重传次数), Y 轴代表平均检测时间. 从图 4 中可看出, 随着阈值 k 的增长, 两种检测方法的平均检测时间都呈现增长趋势, 并且心跳发送间隔大, 检测时间增加. 但是, 共享方法的平均检测时间低于非共享方法的平均检测时间. 而且, 随着阈值 k 的增大, 共享方法的平均检测时间的改进愈明显 (当 $k = 6$ 以及 $\tau = 0.5$ s 时, 平均检测时间的改进是 40%; 当 $k = 6$ 以及 $\tau = 1$ s 时, 平均检测时间的改进是 45%). 这种改进是由于在检测结果共享机制下, 各节点的检测周期是不同步的, 通过共享获得的节点失效的信息可能会早于本地发起的检测, 因而共享方法改进了平均检测时间. 通过实验也证明, 采用检测结果共享机制, 可以有效地降低节点失效的检测时间.

3.2 检测准确性

图 5 显示了在不同的心跳发送间隔 ($\tau = 0.5$ s 和 $\tau = 1$ s) 情况下, 共享方法和非共享方法平均错误发生概率的比较. X 轴代表阈值 k , Y 轴代表平均错

误发生概率. 从图中可看出,随着阈值 k 的增长,两种检测方法的平均错误发生概率都呈现降低的趋势,并且心跳发送间隔愈大,平均错误发生概率愈低. 这是因为通过重传检测消息可很好地改进由于心跳消息丢失造成的失效检测器平均错误发生概率的增加. 当给定阈值 k 较小时,共享方法的平均错误发生概率要高于非共享方法的平均错误发生概率. 但是,当阈值 k 逐渐增大,两种方法的平均错误发生概率差距愈小. 这是由于共享方法可通过其他节点的检测结果去推断目标节点的状态,而在重传次数较大时受心跳消息丢失的影响愈小,从而降低了平均错误发生概率. 在 $k = 6$ 及 $\tau = 0.5 \text{ s}, \tau = 1 \text{ s}$ 时,两种方法的平均错误发生概率几乎一致,而检测时间的差距分别为 46% 和 44%. 根据平均错误发生概率的定义,当 $k = 6$ 及 $\tau = 0.5 \text{ s}$ 时,失效检测器 VC-FD 平均每 6 667 s 发生一次错误检测,已经可满足大部分用户或应用的需求.

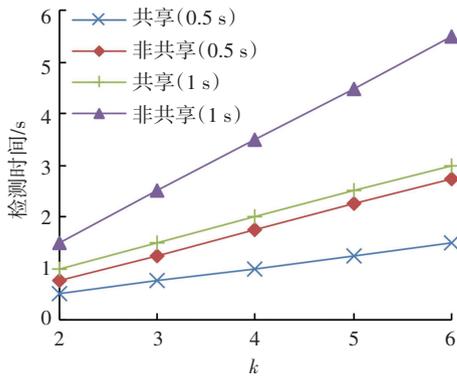


图 4 平均检测时间

Fig.4 Average detection time

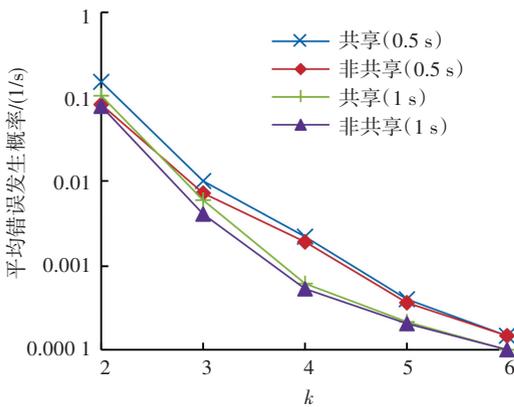


图 5 平均错误发生概率

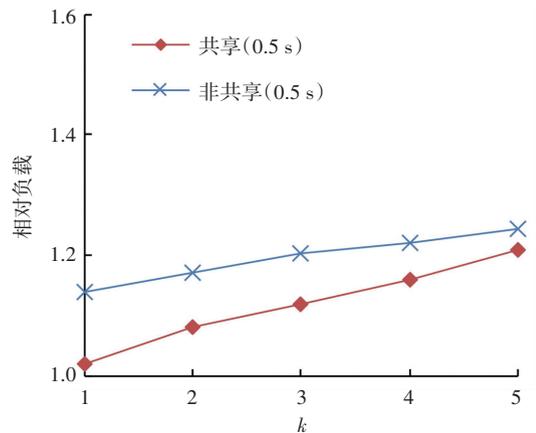
Fig.5 Mistake rate

3.3 检测负载

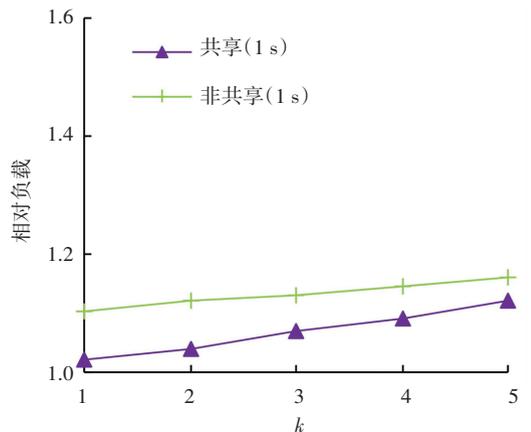
通过测量节点以检测为目的每秒发送的控制包的数量来衡量节点检测负载. 对于非共享方法而言,记录节点心跳消息包的数量. 而对于共享方法而言,除了记录节点心跳消息包的数量,还需要记录

节点通知消息包的数量. 此外,利用相对负载来说明两种检测方法所消耗的检测负载. 相对负载是共享方法所消耗的检测负载与非共享方法所消耗的检测负载的比值.

图 6(a)、(b) 显示了不同心跳发送间隔 ($\tau = 0.5 \text{ s}$ 和 $\tau = 1 \text{ s}$) 两种检测方法的相对负载的比较. 从图中可看出,当心跳发送间隔较小时,两种检测方法分别消耗了更多的检测负载;共享方法的检测负载高于非共享方法的负载,但是随着阈值 k 的增大,两种方法的检测负载的差距在减小. 这是由于当阈值 k 较小时,平均错误发生概率较高,会发送更多的通知消息,所以此时的共享方法会消耗很多的检测负载. 而当阈值 k 较大时,平均错误发生概率降低,被发送的通知消息减少,所以此时的共享方法消耗的检测负载减少. Hayashibara 等^[19] 测试了失效检测器对节点性能的影响,结果显示,运行失效检测器的节点 CPU 负载几乎为常数,远远低于 CPU 满载情况. 失效检测器 VC-FD 在 $\tau = 0.5 \text{ s}$ 以及 $k = 1$ 时,较非共享失效检测器增加了最多约 10% 的节点检测负载. 所以失效检测器增加的节点检测负载不会对节点性能产生影响,是在可接受的范围之内.



(a) 发送间隔为 0.5 s



(b) 发送间隔为 1 s

图 6 相对检测负载

Fig.6 Relative detection overhead

通过以上实验结果,可发现当 $k = 5$ 以及 $\tau = 0.5$ s 的共享检测方法的检测时间、检测准确性要优于 $k = 4$ 以及 $\tau = 0.5$ s 的非共享检测方法. 当 $k > 1$ 时,总能找到合适的阈值 k 和发送间隔 τ 使共享检测方法的检测时间以及平均错误发生概率同时优于非共享检测方法. 而且,在所有的场景中,增加的检测负载没有超过 10%.

4 结 论

失效检测器在可靠的车辆云计算系统中扮演了非常重要的角色. 在本文中提出了基于车辆云计算架构特征的共享失效检测器 VC-FD. 通过对车辆云计算系统进行分层,以及利用 RSU 对车辆节点进行分组,采用检测结果共享机制的 VC-FD 能够适应系统的高度动态性. 通过对比实验,结果显示 VC-FD 在保证准确性的基础上,通过增加有限的检测负载能够有效地改善检测速度. 因此,VC-FD 适合于部署在车辆云计算中提供失效检测服务.

参考文献

- [1] WHAIDUZZAMAN M, SOOKHAK M, GANI A, et al. A survey on vehicular cloud computing [J]. *Journal of Network & Computer Applications*, 2014, 40(1): 325–344. DOI: 10.1016/j.jnca.2013.08.004.
- [2] GU Lin, ZENG Deze, GUO Song. Vehicular cloud computing: a survey [C]// *Proc 2013 Globecom Workshops*. Atlanta, USA; IEEE Press, 2013: 403–407. DOI: 10.1109/GLOCOMW.2013.6825021.
- [3] YU Rong, HUANG Xumin, KANG Jiawen, et al. Cooperative resource management in cloud-enabled vehicular networks [J]. *IEEE Transactions on Industrial Electronics*, 2016, 62(12): 7938–7951. DOI: 10.1109/TIE.2015.2481792.
- [4] REFAAT T K, KANTARCI B, MOUFTAH H T. Virtual machine migration and management for vehicular clouds [J]. *Vehicular Communications*, 2016, 4: 47–56. DOI: 10.1016/j.vehcom.2016.05.001.
- [5] GHAZIZADEH P, MUKKAMALA R, EL-TAWAB S. Scheduling in vehicular cloud using mixed integer linear programming [C]// *Proc 1st International Workshop on Mobile Sensing, Computing and Communication*. Philadelphia, USA; ACM Press, 2014: 7–12. DOI: 10.1145/2633675.2633681.
- [6] WEI Wei, FAN Xunli, SONG Houbing, et al. Imperfect information dynamic stackelberg game based resource allocation using hidden markov for cloud computing [J]. *IEEE Transactions on Services Computing*, 2016: 1–13. DOI: 10.1109/TSC.2016.2528246.
- [7] SHOJAFAR M, CORDESCHI N, and BACCARELLI E. Energy-efficient adaptive resource management for real-time vehicular cloud services [J]. *IEEE Transactions on Services Computing*, 2016, 7: 1–1. DOI: 10.1109/TSC.2016.2551747.
- [8] PANNU H S, LIU Jianguo, GUAN Qiang, et al. AFD: adaptive failure detection system for cloud computing infrastructures [C]// *Proc 31st International Performance Computing and Communications Conference*. Austin, USA; IEEE Press, 2012: 71–80. DOI: 10.1109/PCCC.2012.6407740.
- [9] XIONG Naixue, VASIAKOS A V, WU Jie, et al. A self-tuning failure detection scheme for cloud computing service [C]// *Proc 26th International Parallel and Distributed Processing Symposium*. Shanghai, China; IEEE Press, 2012: 668–679. DOI: 10.1109/IPDPS.2012.126.
- [10] WANG Pu, ZHENG Jun, LI Cheng. Cooperative fault-detection mechanism with high accuracy and bounded delay for underwater sensor networks [J]. *Wireless Communications & Mobile Computing*, 2009, 9(2): 143–153. DOI: 10.1002/wcm.591.
- [11] HSU Chinjung, CHUNG Wuchun, LAI Kuanchou, et al. Cooperative failure detection in multi-overlay environments [J]. *Journal of Internet Technology*, 2011, 12(2): 259–267. DOI: 10.6138/JIT.2011.12.2.07.
- [12] PITREY C, SAILHAN F. Revisiting gossip-style failure detection in wireless sensor network [C]// *Proc 32nd International Conference on Computer Safety, Reliability and Security*. Toulouse, France; HAL Press, 2013.
- [13] OLARIU S, KHALIL I, ABUELELA M. Taking VANET to the clouds [J]. *International Journal of Pervasive Computing & Communications*, 2011, 7(1): 7–21. DOI: 10.1108/1742737111123577.
- [14] ARIF S, OLARIU S, WANG Jin, et al. Datacenter at the airport: reasoning about time-dependent parking lot occupancy [J]. *IEEE Transactions on Parallel & Distributed Systems*, 2012, 23(11): 2067–2080. DOI: 10.1109/TPDS.2012.47.
- [15] ATALLAH R, ASSI C, YU Jiayuan. A reinforcement learning technique for optimizing downlink scheduling in an energy-limited vehicular network [J]. *IEEE Transactions on Vehicular Technology*, 2016: 4592–4601. DOI: 10.1109/TVT.2016.2622180.
- [16] FELBER P, DEFAGO X, GUERRAOU R, et al. Failure detectors as first class objects [C]// *Proc 1st International Symposium on Distributed Objects and Applications*. Edinburgh, UK; IEEE Press, 1999: 132–141. DOI: 10.1109/DOA.1999.794001.
- [17] CHEN Wei, TOUEG S, AGUILERA M K. On the quality of service of failure Detectors [J]. *IEEE Transactions on Computers*, 2002, 51(1): 13–32. DOI: 10.1109/12.980014.
- [18] DURRETT R. *Probability: theory and examples* [M]. London: Cambridge University Press, 2010: 81–86.
- [19] HAYASHIBARA N, DEFAGO X, YARED R, et al. The φ accrual failure detector [C]// *Proc 23rd IEEE International Symposium on Reliable Distributed Systems*. Florianopolis, Brazil; IEEE Press, 2004: 66–78. DOI: 10.1109/RELDIS.2004.1353004.

(编辑 苗秀芝)