传感器网络调试研究综述

马峻岩 周兴社 张 羽 李士宁 李志刚
(西北工业大学计算机学院 西安 710072)

摘 要 越来越多面向不同应用领域的传感器网络被部署在真实环境中,帮助人们以新的方式观测周围的物理世界.然而,这些系统常常会出现各种不可预期的故障,能否快速有效地对这些故障进行检测、定位和修复,是传感器网络调试需要研究的重要内容.在概述了传感器网络调试问题之后,文章总结、比较了传感器网络调试过程中常用的系统状态信息获取技术,然后从故障检测、故障定位和故障修复3个方面综述了代表性关键技术及相关工具,最后探讨了该领域未来的研究方向.

关键词 传感器网络;调试;故障检测;故障定位;故障修复;物联网
中图法分类号 TP311 **DOI号**: 10.3724/SP.J.1016.2012.00405

Debugging Sensor Networks: A Survey

MA Jun-Yan ZHOU Xing-She ZHANG Yu LI Shi-Ning LI Zhi-Gang
(*School of Computer Science, Northwestern Polytechnical University, Xi'an* 710072)

Abstract Various sensor networks have been increasingly deployed in the real world to help us observe our surrounding environment. However, it is common that the deployed sensor networks suffer from unpredictable faults. Hence, fast and effective fault detection, localization and repair are of utmost importance for sensor network debugging. Extraction techniques of system state information for sensor network debugging are summarized and compared after a brief introduction of debugging sensor networks. Representative debugging techniques and related tools are then surveyed from view of fault detection, localization and repair. The future research is discussed at the end of this paper.

Keywords sensor networks; debugging; fault detection; fault localization; fault repair; Internet of Things

1 引 言

传感器网络是实现物联网的重要基础[1],其广泛获取客观物理信息的能力,使它在环境监测、军事侦察、医疗卫生、智能人居环境及抢险救灾等领域具有十分广阔的应用前景.随着技术的进步和研究的深入,越来越多面向不同应用领域的传感器网络被部署到各种真实环境中,帮助人们以新的方式观测周围的物理世界[2-5].然而,这些在真实环境中运行的系统常常会出现各种不可预期的故障[6].这些故障有的可能仅仅是由于节点封装保护不当所致,但另外很大一部分产生的原因则与传感器网络这类新型感知计算方式本身的一些特性密切相关.能否快

收稿日期:2011-08-22;最终修改稿收到日期:2012-01-06.本课题得到国家科技支撑计划项目(2007BAD79B00)资助.**马峻岩**,男,1982年生,博士研究生,主要研究方向为传感器网络故障诊断与调试.E-mail: alexmajy@gmail.com. **周兴社**,男,1955年生,教授,博士生导师,主要研究领域为嵌入式计算与普适计算. **张 羽**,男,1975年生,博士,副教授,主要研究方向为传感器网络与嵌入式操作系统. **李士宁**,男,1967年生,教授,博士生导师,主要研究领域为传感器网络与移动计算. **李志刚**,男,1975年生,博士,副教授,主要研究方向为网络化嵌入式技术与传感器网络.

速有效地对这些故障进行检测，确定故障形成的原因并加以修复，是传感器网络需要解决的关键问题之一[7].

调试主要涉及故障的检测、定位与修复.本文综述传感器网络调试技术的研究进展，具体结构安排如下：第 2 节概述传感器网络调试问题；第 3 节介绍故障检测与定位过程中涉及的主要系统状态信息，并对常用的系统状态信息获取技术进行总结与比较；第 4 节综述当前具有代表性的故障检测、定位和修复技术；第 5 节展望未来的研究方向；第 6 节对全文进行总结.

2 传感器网络常见故障及评价准则

2.1 常见故障及分类

本文中传感器网络故障是指系统运行过程中出现的不希望或不可接受的行为.按照系统层次关系，我们可以将传感器网络故障从高到低，分为应用故障、网络故障和节点故障三类，表 1 对各类常见故障进行了总结.这里低层次故障通常会以某种形式在高层次故障中反映，而不同低层次故障可能会表现为同一高层次的形式.例如，链路失效可能是节点损毁引起，也有可能是错误软件行为所致；感知数据缺失可能是网络大量丢包造成，也有可能是路由中存在环路的结果.

表 1 传感器网络常见故障及分类

应用故障	网络故障	节点故障
感知数据缺失，检测延时较长，生命期过短	链路失效，大量丢包，网络拥塞，路由环路，网络断裂	重启，损毁，硬件故障，无响应，能量过早耗尽，传感器读数故障，错误软件行为

这些故障产生的原因主要包括环境干扰、硬件失效和软件失效.由环境干扰和硬件失效产生的故障一般可以通过容错技术消除故障的影响，而软件失效产生的故障，只有在软件缺陷被修复后才可能不会继续对系统运行造成影响.软件失效引发的故障往往因系统资源和部署环境的限制而难以调试，本文重点研究这一类故障的调试问题.

2.2 问题与挑战

作为一种新型的感知计算方式，传感器网络具有鲜明的特点，而这些特点也为调试提出了一系列挑战.

(1) 与物理世界紧密耦合

传感器网络被部署在物理世界中，在无人看护的情况下感知环境状况(如温度、空气质量、震动等)，对感知数据进行处理并实时地做出反应.

由于与物理世界紧密偶合，传感器网络的功能和行为在很大程度上受到部署环境影响.这就导致部署前对系统进行的各种测试，仅能对实际部署后系统的正确性与各项性能参数做出一个大致评估，那些部署后出现的问题常常无法在部署前的测试中发现.此外，有些部署环境使调试工作仅能以远程方式进行，这也为传感器网络调试带来了不少困难.

(2) 资源高度受限

受成本、体积和功耗的限制，传感器节点的计算能力、存储空间和通信带宽都比传统计算方式中的节点(如服务器、个人 PC、手持移动设备和专用嵌入式设备等)要弱很多.此外，由于节点一般靠电池驱动，而很多情况下受环境和节点数量的限制，更换电池往往不易实现，这使能源成为系统中最为宝贵的资源.

系统状态可见度是指调试人员观测节点程序内部状态的能力，是实施调试的重要前提[8-10].然而可见度和资源开销之间存在着根本的矛盾，外部观测者通过与节点之间的通信获得节点内部状态，增加可见度意味着将引入更多的资源开销.传感器网络资源高度受限为故障的检测、定位和修复带来了巨大的挑战.

(3) 应用相关性

与构建在通用网络协议栈上的传统分布式系统不同，传感器网络要根据应用在功能、性能、部署方式和部署环境等方面的需求，对通信协议、能量管理和任务调度等系统各个部分进行定制化设计.

定制化设计满足了应用在多方面的需求，然而缺乏通用基础软件的支持，意味着传感器网络将缺少坚固的固件(firm firmware[11]).由于测试环境与部署环境的差异，再加上适用于传感器网络测试方法与工具的欠缺，传感器网络部署前往往缺乏足够的测试.因此部署后的系统更容易受到各类故障的影响.

(4) 持续运行

感知数据的获取是支撑上层应用的关键，而像自然环境监测、电网设备的运行管理以及水资源质量控制等应用，需要对相关对象或过程进行不间断地监测.因此传感器网络需要长期持续运行，为上层应用提供感知数据.

故障可能发生在系统运行过程中的任何阶段，有时为了保证传感器网络持续运行，需要避免调试

时的探针效应(probe-effect)，即要在尽量不影响或中断系统运行的前提下实施系统调试. 此外，长期持续运行还会使传感器网络会受到“软件衰退”的影响[12-13]. 这些都对受资源以及部署环境限制的传感器网络调试提出了更大的挑战.

2.3　调试技术的评价准则

传感器网络调试中，人们比较关注的评价准则包括误报率、漏报率、资源开销和速度等.

(1) 误报率和漏报率. 误报是指当系统不存在某种故障时，故障检测报告系统可能存在某种故障. 误报率等于误报数除以故障检测总数. 漏报是指系统中存在某类故障，而故障检测没有报告该故障. 漏报率等于漏报数除以故障检测总数. 误报率和漏报率会受很多因素的影响，如检测模型以及检测信息质量等. 过多误报会导致大量的人工分析，而过多漏报则会使故障检测有效性降低. 故障检测时的误报率和漏报率越小，则表明故障检测越好. 然而，误报率和漏报率之间往往存在着折中关系，即有时降低误报率会使漏报率增加，反之有时降低漏报率也会导致误报率升高.

(2) 资源开销. 传感器节点资源高度受限，当调试需要占用节点上各类资源时，这些资源的开销将成为衡量调试技术好坏的另一类重要评价准则. 相关资源开销包括通信开销、存储开销(包括数据存储和程序存储)、计算开销和能耗开销. 由于无线通信是传感器网络能耗开销的主要因素，因此能耗开销一般借助通信开销间接的衡量. 系统运行时的状态信息是故障检测和定位的重要依据，这些状态信息的获取是资源开销的主要来源. 在有限的资源开销与足够的状态信息之间进行权衡，是传感器网络故障检测和定位需要解决的一个主要问题. 当通过代码更新进行故障修复时，则需要尽可能减少更新中的通信开销.

(3) 速度. 如果调试涉及大量计算(这里通常是指非传感器节点上的计算)，则速度也是衡量某一调试技术好坏的一个重要准则. 例如，在使用静态分析技术对传感器网络程序进行故障检测时，分布式系统中的非确定性会导致状态空间爆炸问题，一个好的故障检测方法需要在保证较低误报率和漏报率的前提下，对状态空间进行有效化简，提高故障检测的速度.

3　系统状态信息的获取

系统运行过程中的状态信息是实施故障检测和定位的重要依据，而系统资源上的限制为传感器网络状态信息获取提出了挑战. 状态信息获取包括信息的输出与收集. 这一节我们首先介绍传感器网络中常用的系统状态信息，然后从状态信息的输出和收集两方面对现有相关研究进行总结与比较.

3.1　系统状态信息

传感器网络系统状态包括单个节点状态和节点之间的通信状态，其中单个节点状态可进一步分为硬件状态与软件状态.

3.1.1　硬件状态

硬件状态涉及组成节点的各个元器件(如处理器、射频芯片和传感器等)，主要包括这些器件的工作状态以及可能受环境影响而变化的相关特性参数等. 由于节点能耗相对容易测量，且能反映各个器件工作状态的综合情况，因此成为一种常用的系统状态信息. 表 2 根据数据手册给出了 IRIS 节点① 主要器件不同工作状态组合时的综合能耗情况，可以看出利用节点详细的能耗记录，调试人员可以对节点行为和状态进行分析与推断.

表 2　IRIS 节点主要器件不同状态时节点能耗情况

MCU	Radio	Flash	电流/mA
sleep	off	off	0.001
on	off	off	0.9
on	off	read	10.9
on	off	write	17.9
on	rx	off	16.9
on	tx	off	17.4

3.1.2　软件状态

在运行过程中，节点存储器和寄存器中所有时刻的内容，包含了软件执行时的主要状态信息. 根据程序不同的构建方式，这些内容可以进一步被映射为软件不同抽象层次上的各种状态视图，例如程序计数器的变化反映了机器代码的执行情况，而这些机器代码又对应着上层软件抽象中的不同部分. 中断的执行、操作系统中任务的调度、程序中变量的修改以及函数的调用执行，这些视图从不同的层面和视角反映了软件的执行情况，包含了故障检测和定位所需的关键信息.

3.1.3　通信状态

通信状态包括节点之间消息交换的内容和交换发生的时间. 节点通过消息传递进行协作，完成特定任务，通信状态影响了节点自身状态以及节点间的交互. 例如，网络中簇首的选举、邻居关系的确定以

① IRIS, wireless modules. http://www.memsic.com

及下一跳路由的选择等都会受到通信状态的影响.通信状态决定了网络中节点之间的关系,这些关系一方面可以通过分析节点之间交换的消息进行推断,另一方面也可以从程序中存储的邻居表和路由表等相关数据结构获得.

3.2 系统状态信息的输出

系统状态信息输出主要包括正常输出、平台接口输出和插装代码输出三种方式[14].

3.2.1 正常输出

传感器网络在运行过程中会产生一系列输出.这里正常输出是指:系统运行过程中非调试目的的、必要的信息输出,如节点上报的感知数据以及网络中的路由维护消息等.这些正常输出信息,可以用于系统状态的推断,如 Suelo[15] 使用节点上报的传感器数据对传感器故障进行检测,SNIF[8] 和 LiveNet[16] 则通过无线监听得到的网络消息,对网络故障和性能进行检测与分析.然而,受资源上的约束,传感器网络运行过程中的正常输出往往非常有限.在不借助无线监听和其他信息输出方式时,通常很难对传感器网络进行调试.针对这一问题,有研究人员提出使用可见度(visibility)准则进行传感器网络协议设计[17-19],以克服使用正常输出进行调试的局限.

3.2.2 平台接口输出

为了便于调试,一般系统都向用户提供获取额外系统状态信息的接口,如 Linux 的 ptrace 系统调用以及 Java 虚拟机的 JVM TI 接口等.然而,受资源以及运行时修改能力的限制,大多数传感器节点操作系统对这方面的支持较少,且提供的接口功能也十分有限.例如,TinyOS[20] 仅提供 CTP(Collection Tree Protocol)协议[21] 调试信息的获取接口 *CollectionDebug*;Contiki[22] 的远程 Shell 则允许用对节点运行的线程、能量消耗和简单的网络通信情况进行查询,类似的还有 BTNut[23] 和 LiteOS[24] 提供的 Shell 接口.通过这些接口,用户可以获取一些简单的调试信息.

3.2.3 插装代码输出

正常输出和平台接口提供的调试信息非常有限,特别是在故障定位时,用户往往需要更多相关的系统内部状态信息对故障位置进行分析.插装代码可以帮助用户获得如变量状态、函数调用记录以及堆栈使用情况等各类系统内部状态,为深入分析故障提供帮助.

(1)源码插装.源码插装是指通过手工或者自动方式修改程序源代码,在需要实施监测的位置增加相关信息输出语句(如改变 LED 状态、输出信息到串口或增加日志记录等),例如在日志中记录某一变量每次被修改的数值.该方法在实现上相对简单且较为灵活,但插装后需要对源代码进行重新编译,这对部署后系统意味着需要更新节点上的二进制代码.更新开销过大和节点重启所导致的相关程序状态丢失,是代码更新的两个主要问题.代码执行方式决定了代码更新时的数据传输量,但同时也会限制插装方法可获得的调试信息类型,即代码修改能力.例如,虽然虚拟机[25-26] 上运行的字节码程序相对体积较小,但这些程序的插装却无法获得底层虚拟机相关的状态信息.此外,如果代码更新涉及到 ROM 的擦写,还会受到 MCU 编程电压的限制.依靠普通电池供电的节点电压会随系统运行时间的增加而逐渐降低,当电压低于某一阈值后,ROM 擦写将无法实施.最后,如果代码更新需要重启节点,则可能造成重要调试信息的丢失(如故障发生时错误的路由状态信息),从而增加了调试难度与周期[2].本文 4.3.1 节详细讨论了几类主要代码更新方式在修改能力、更新开销和是否需要重启等方面的特性.

(2)目标码插装.目标码插装是在不修改源代码的情况下对程序实施插装的方法.该方法主要用于对运行程序的动态修改,也可用于一些无法实施源码插装的情况,例如对无源码的二进制库进行插装.在目标码中插入新指令的同时,保证原目标码中指令对地址以及数据的引用依然有效是该方法实施时需要解决的关键问题.Trampoline 是目标码插装的一项常用技术[27-28].假设插装点程序地址为 A,相应的指令为 I,则插装时指令 I 将被替换为指向程序中某一特定区域的跳转指令.特定区域包含了相关状态的提取代码和被替换的指令 I,该区域最后的跳转指令将指向原地址 A 处指令 I 的下一条指令地址,即 A+sizeof(I).这种方式使插装代码在不改变原先代码逻辑的前提下,可以获取程序运行时的相关状态信息.动态目标码插装避免了源码插装中代码编译和更新的步骤,同时也防止了因重启而导致的状态丢失.但由于涉及机器指令的修改,与源码插装相比,该方法实现时较为复杂且通用性差.与源码插装类似,如果目标插装涉及到 ROM 改写,则还会受到 MCU 编程电压的限制.

(3)调用截取.调用截取是通过调用者与被调用者之间安插的截取器,获取二者间的传递消息,并完成特定处理工作的一种插装方法,常被用于性能

参数收集和调试等领域[14]. 由于调用截取方法通常不需要对目标程序进行修改，与其他方法相比其代码入侵性较弱，但仅能获取和修改函数之间传递的信息. 对于支持运行时模块动态加载的传感器节点操作系统如 Contiki[22] 和 SOS[29] 等，动态调用截取可以通过系统提供的运行时链接实现[30]. 当操作系统不具备这样的功能时，也可利用目标码插装加以实现[31]，但较前者复杂.

3.3　系统状态信息的收集

传感器网络资源以及部署环境的限制为调试时的状态信息收集带来了挑战，可使用的收集方式极其有限. 常见的收集方式有以下四种：带内收集、本地存储、网络监听和带外收集[9]. 表 3 对不同收集方式的特性进行了总结.

表 3　不同系统状态信息收集方式的对比

	数据带宽	干扰	灵活性	附加连接	额外硬件
带内收集	低	高	高	否	否
本地存储	高	低	高	否	否
网络监听	低	低	中	否	是
带外收集	高	低	低	是	是

3.3.1　带内收集

带内收集利用传感器网络自身无线传输带宽将收集的数据发送回基站. 带内收集可以使用应用程序已有的路由，但有时为了避免应用失效对数据收集的影响，也可以使用独立于应用的收集协议[32]. 带内收集不需要额外基础设施或硬件的支持，是最常用有时甚至是唯一的收集方式，但这种方式会占用传感器网络的传输带宽并可能对应用产生影响. 因此在使用带内收集方式时，往往尽可能的通过聚合[33]、捎带[34]、分布式处理[35-36]以及优化轻量级监测方案[37]等手段减少带内传输开销.

3.3.2　本地存储

本地存储将收集的数据保存在传感器节点 Flash 存储器中，供事后提取与分析. 与带内收集相比，本地存储在信息记录过程中不产生额外的通信开销，所以不会对应用通信造成太大影响. 其调试信息收集主要受到存储器容量的限制. 对于本地存储信息，其处理方式可分为集中和分布式两种：集中式处理需要将节点 Flash 中的信息，通过串口或者无线方式全部提取后进行集中式的处理. 串口提取方式不会对网络通信造成影响，但要求节点是物理可接触的；无线方式提取时，可以使用独立信道或者移动基站减小数据提取过程对网络通信的影响，然而过多的数据仍会产生过大的通信开销. 分布式处理将部分调试分析工作交由节点完成[38]，克服了集中式处理数据提取开销过大的问题. 随着低功耗、大容量存储设备在传感器节点上应用和普及[39]以及节点上功能更加强大的轻量级文件系统如 Coffee[40] 和数据库的出现[41]，利用节点本地存储和计算能力进行分布式的调试将成为未来传感器网络调试的一个研究热点.

3.3.3　网络监听

网络监听借助无线信道共享特性，使用额外部署的监听节点(sniffer)进行数据收集. 该技术最早被应用于 WLAN 的监测与分析[42]，可以在最大程度上减少数据收集过程对原有系统的影响. 根据监听节点能力的不同，监听可分为在线[8]和离线[16]两种模式. 在线监听节点一般配备能力较强且不会对传感器网络通信造成干扰的通信模块(如 WLAN 或蓝牙等)，可以将监听到的数据实时传送回数据处理中心. 离线监听节点与普通节点类似，将监听到的消息存储到本地 Flash 中供事后分析，但用户需要解决监听节点的回收或监听数据的提取问题. 除了网络通信时产生的正常输出外，用户还可以让节点利用带内通信以广播的形式向监听节点发送程序内部状态，这种方法与带内收集方式相比具有更小的开销，且调试时对系统的影响较小[9]. 采用监听方式进行数据收集时，关键要解决多个监听节点记录的同步问题以及监听丢包导致的非确定状态的推断问题[9,43].

3.3.4　带外收集

带外收集是指利用节点无线传输以外方式(如串口)进行的数据收集，该方式一般需要进行节点硬件上的附加连接或修改. 与其他方式相比，带外收集一般具有较高的数据传输带宽，且不会影响应用的通信. 带外收集是测试台(Testbed)中最常见的数据收集方式[44-46]. 除了测试台，带外传输也可用于部署后系统状态信息的收集. 例如，Khan 等人[47] 使用附加的低成本无线传输模块收集节点能耗使用情况，并利用这些信息对节点能量耗尽、天线损坏、系统崩溃和异常重启等故障进行检测.

4　关键技术

4.1　故障检测

故障检测是故障定位和修复的前提，主要包括数据故障检测、网络故障检测和软件故障检测三类.

4.1.1　数据故障检测

作为数据型网络，感知数据的正确性对传感器

网络应用尤为重要.为了获取物理世界信息,传感器网络通常被部署在无人看护的户外或者环境较为恶劣的区域,这使得感知数据质量很容易受到部署环境的影响.如果传感器上报数据与实际物理现象不一致,我们则认为发生了数据故障[48].硬件故障、连接失效、校准参数的漂移、环境噪音以及软件缺陷都会导致数据故障.故障数据的处理和传输会浪费系统资源,更重要的是大量故障数据会导致应用本身无法正常工作.数据本身的属性(如最大值、最小值和变化率等)以及多个数据点在时空上的约束关系(如同一时刻相邻位置或者同一位置相邻时刻的温度具有相关性)是数据故障检测的主要依据.

分布式数据故障检测可以减少因错误读数传输产生的额外开销,但检测算法复杂度会受传感器节点资源限制.Gao 等人[49]通过节点数据与邻居节点数据的加权中值比较进行故障检测,并根据检测结果更新权值提高检测方法对动态故障的适应性.Rajagopal 等人[50]则提出利用节点与邻居感知数据在相关性上的突变进行数据故障检测.类似的分布式数据故障检测研究还有很多,但这些研究大部分没有考虑故障检测算法在实际节点上应用的开销,因此适用范围也具有一定的局限性.

Ni 等人[48]认为虽然分布式处理有助于减少总通信开销,但融合中心相关信息减少,会导致故障决策可信度降低.此外,由于集中式检测可以借助大量历史数据以及高性能计算设备对数据进行处理和分析,因此 Ni 等人认集中式数据故障检测较分布式更具实用性.Sharma 等人[51]针对文献[48]中总结的传感器网络典型数据故障提出了基于规则、基于估计、基于学习以及混合等 4 种具有代表性的集中式故障检测方法.实验表明在中高度故障率情况下,这 4 种方法均具有较高的检测效力,而其他情况下的检测效力则依赖于故障的类型.为了解决因土壤传感器不可靠导致的读数或者校准失效问题,Suelo[15]首先将感知数据变换为故障特征空间中的向量,然后依据已知故障数据库向用户指示需要采取的相关措施(如校准、验证读数等),最后用户根据传感器的实际情况对故障数据库进行更新,从而达到对故障检测系统改进与优化.FIND[52]则通过将多个节点感知数据实际测量结果排序,并将该排序与根据节点位置关系得到的理论排序相对比,对数据故障进行检测.

4.1.2 网络故障检测

传感器网络常用于收集分布于空间与时间上的各种物理世界信息,网络的正常与否直接影响到这些信息的收集.网络故障检测重点研究网络通信中如链路失效、网络拥塞和节点失效等相关问题的检测.这里网络故障检测是从网络通信的角度出发解释系统不能完成某项数据传输任务的原因,也就是常说的网络故障诊断.例如,网络故障检测可以确定数据丢包是由某些节点异常行为导致的网络拥塞所造成的,但一般不关心导致这些节点异常行为的软件缺陷是什么.正如 2.2 节所讨论的那样,应用相关产生的定制化设计以及环境对系统的影响,使传感器网络的网络通信更容易受到软件缺陷的影响,因此网络故障检测与调试密切相关.网络故障检测主要可分为主动监测和被动监听两类.

(1)基于主动监测的故障检测

有限的能量和传输带宽,使得基站接收的感知数据往往成为网络故障检测仅有的依据,然而这些数据提供的信息十分有限.主动监测通过收集额外的网络通信参数解决网络故障检测信息不足的问题.

Sympathy[53]根据传感器网络数据通信特点,提出了一组与网络连通性、网络数据流和节点相关的监测参数用于网络故障检测.利用收集到的参数以及相关决策树算法,用户可以对几种常见的网络故障进行检测.该方法简单实用,适用于大部分数据收集类应用,可检测网络中节点的失效、重启、邻居缺失和路由失效等故障,但缺点是相关参数收集产生的通信开销较大.

针对 Sympathy 故障检测开销过大的问题,Wachs 等人[19]提出从协议设计出发,减少网络故障检测时的可见性开销.Wachs 等人将可见性开销定义为对每种故障发生概率与检测出该故障所需能耗开销乘积的和.通过对协议的重新设计,减少协议的可见性开销,实现对故障检测开销的优化.然而,重新设计协议的方法并不一定适用于大多数情况.

针对数据收集类应用,PAD[19]提出了一种轻量级的故障检测方法.该方法首先利用轻量级数据包标记策略,得到网络拓扑和一些初步网络运行信息(如网络拓扑和路由相关信息),其次根据这些初步的结果得到网络元素的依赖关系图和推断模型,然后使用观测到的症状作为推断模型输入,计算在某一故障条件下,网络中不同元素发生故障的后验概率,最后通过各种故障后验概率的比较实施故障检测.

基于特定模型的分析方法通常只能处理已知的故障,AD[54]提出借助相关图(correlation graph)进行故障检测,其中相关图描述了节点各个参数在一

段时间内的统计上互相关性.文章认为在网络正常运行情况下,同一节点不同时刻的相关图具有时间相关性,而同一时刻不同节点的相关图具有空间相关性.AD 通过相关图在时间与空间上的突变检测实现网络故障的检测.该方法开销较小,适用于数据收集类应用的网络故障检测,但可能并不适用于应用场景较为复杂的情况(如目标跟踪).

(2)基于被动监听的故障检测

利用无线传输介质共享的特性,被动监听可以在完全不影响应用本身的情况下,收集用于故障检测的信息.在允许部署额外监听节点的情况下,监听节点收集的各种网络通信消息可以为网络的性能分析与故障检测提供足够的信息.

SNIF[8]允许用户利用监听到的数据包对监测类传感器网络应用的网络故障进行检测.根据监听到数据包中的特定字段,SNIF 可以对节点死亡和重启、邻居缺失、路由环路、网络断裂等故障进行检测.例如,数据包中序列号(sequence number)的重置表明节点重启,一段时间内链路广播中不包含任何邻居信息,则表明节点邻居缺失.此外,用户还可以利用 SNIF 提供的面向数据流的分析框架实现自定义的故障检测.基于监听的网络故障检测的类似研究还有很多[16,43,55],其区别主要体现在监听数据获取与处理方式上的不同.

4.1.3　软件故障检测

传感器网络中很大一部分故障与软件缺陷有关,如果能在软件层面尽早检测出故障,则会减少后期故障定位的难度.因此有效的软件故障检测技术可以在很大程度上解决传感器网络故障定位问题.

(1)基于测试的故障检测

测试是系统部署前故障检测的主要手段,测试环境的相对可控性,使部署前的故障检测较部署后更为容易.然而,测试环境与实际部署环境的差异以及现有测试方法在应用于传感器网络测试时的各种不足,很大程度上降低了部署前测试对软件故障检测的有效性.针对这一问题,基于测试的故障检测一方面需要尽可能减小测试环境与部署环境的差异,另一方面需要根据传感器网络特点对传统测试方法进行改进.

表 4　不同测试环境可扩展性与真实度的对比

	类 ns-2 通用模拟器	集成模拟开发环境	指令集精度模拟器	测试台
可扩展性	+++	+++	++	+
真实性	+	++	+++	++++

在测试环境方面,通用网络模拟器如 ns-2① 等可以用于系统设计初期网络协议的性能分析与评价,但通用模拟器存在两点不足.首先,开发人员需要对协议进行二次实现;其次,二次实现代码在实际运行前缺乏有效的验证手段.这大大增加了系统实际运行时出现故障的可能,从而加重后续的系统调试工作.针对通用模拟器的不足,集成模拟开发环境如 TOSSIM 等[56-57],将通用模拟器与系统开发环境进行集成,使用户可以直接使用编写的应用程序代码进行网络模拟.为了屏蔽不同硬件平台的差异,集成模拟开发环境一般是在相关操作系统的某一组 API 上构建模拟器,将应用代码与相关操作系统 API 的模拟库链接实现集成开发环境中的模拟.由于集成模拟开发环境是在某组特定 API 上进行模拟,所以无法完全反映代码在节点上运行的真实情况.指令级精度模拟器直接使用交叉编译生成的目标代码进行模拟[58-59],进而使该环境下的测试可以对因中断等一些时序问题引发的故障进行检测,但指令级精度模拟器的使用受限于特定的节点硬件平台,其通用性较差.虽然模拟器可控的执行环境以及详尽的执行记录,为故障检测带来了很大的便利,但大量真实世界的简化模型(如无线传输和能耗等)使得很多实际部署中的故障无法在模拟器中再现.测试台[44-46]则允许开发人员以相对可控的方式,对由真实节点构成的传感器网络进行各方面性能和功能的测试,增加了系统测试的真实度.表 4 对不同测试环境可扩展性与真实度之间的差别进行了总结.

在测试方法方面,当使用真实节点进行测试时,由于测试可能涉及到多个节点、多类节点硬件平台和不同计算能力的设备(如 PC、节点和移动基站等),因此传统的测试方法与自动化测试框架无法直接应用于传感器网络.TUnit② 针对上述问题,对现有测试软件架构进行扩展,为 TinyOS 设计了自动化的单元测试框架.然而,TUnit 需要用户使用 TinyOS 编写测试用例,且测试断言只能引用节点的本地状态.MUnit[60]则利用 Embedded RPC 技术[61],允许用户在 PC 端使用更加简洁的脚本语言编写测试用例,并对涉及多个节点的分布式状态进行断言.这些技术都在不同程度上简化和改进了传感器网络部署前的测试过程,使开发人员可以将传统测试技术应用于传感器网络开发.然而,应用相关

① ns-2, the Network Simulator. http://isi.edu/nsnam/ns

② TUnit. http://docs.tinyos.net/tinywiki/index.php/TUnit

性导致的定制化设计,使得传统测试方法不能满足传感器网络所有的测试需求.特别是在协议测试方面,现有传感器网络系统开发人员缺乏足够的方法与工具,在真实节点上对定制化的通信协议进行功能与性能方面的充分测试.这一问题已经逐渐引起了人们的关注,目前的研究主要包括通用评测环境的设计[62]、基于可控无线干扰的测试[63-64]以及基于故障注入测试[65]方法的研究.

(2)运行时故障检测

正如本文2.2节所述,传感器网络与物理世界紧密耦合的特点,使得部署前的测试仅能对实际部署后系统的正确性与各项性能参数做出一个大致评估,而那些部署后出现的问题常常无法在部署前测试中发现.部署后故障需要使用运行时故障检测技术进行检测.程序运行时故障检测是根据程序运行时状态,对故障进行自动识别的过程.资源上的限制使传感器网络具有非常有限的可见性,而部署环境的复杂与不可控性又使故障再现十分困难,这些都为部署后的故障检测与定位带来了巨大的挑战.有效的运行时故障检测可以使传感器网络具有fail-fast属性,具有这种属性的系统能在故障发生后尽可能早地捕获故障,并向用户提供与引发故障最为相关的系统状态信息,从而帮助用户更加容易和快速地对故障进行定位[66].根据故障涉及的节点个数,运行时故障检测可分为单节点与多节点两类.

①单节点故障检测

出于效率原因,节点上的程序通常使用C或者类似C的编程语言如nesC[67]进行编写,这使得空指针引用以及数组越界访问等错误往往难以避免.然而,由于节点数据存储空间非常有限,节点上通常没有用户/内核边界和内存保护机制.因此与其他计算机系统相比,缓冲区溢出、栈溢出以及数组越界等运行时故障,对节点的影响更为严重.这类故障一旦发生通常难以调试,节点程序将可能进入不可预知的状态,失去对外界的响应.因此需要对这类故障进行检测,并在故障发生时采取相应的恢复措施.Safe TinyOS[68]基于Deputy[69]编译器提供的标记语言对TinyOS组件进行安全化标记,使程序可以在运行时检测空指针引用以及数组越界访问等故障,同时向用户提供用于定位故障的相关信息.此外,如果在检测到故障后采取重启或微重启技术对节点进行恢复[70],将改善系统的可用性.

TinyOS组件化编程模型最大限度地增加了代码的复用度,很大程度上解决了因应用定制产生的代码复用问题.在TinyOS中,接口定义了组件之间交互的唯一方式,然而原有接口只包含了类型信息,并未给出其精确的语意以及调用模式的规范.在很多情况下,用于连接各类组件的接口在定义时包含了很多隐含的使用限制,这些限制会在组件使用不当时导致程序出现故障.这种问题会随着应用复杂度和代码规模的增加而变得更加严重.当这类故障发生时,用户往往需要深入分析他人的组件实现以确定故障原因.为了解决上述不足,Archer等人[71]为TinyOS接口设计了一种契约(contract)描述语言.开发人员使用该语言定义接口调用时的前件(pre-condition)与后件(post-condition),这些契约最终与源程序一起编译为可执行代码,在程序执行时对接口契约进行检查,并在契约违反时帮助开发人员快速定位故障.当需要检查的接口过多时,保存每个接口状态的数据存储开销是限制该方法应用的主要因素.

②多节点故障检测

传感器网络由多个节点组成,这些节点通过相互协作完成特定的感知任务,软件缺陷可能会使协作失效从而导致故障.由于需要获取多个节点的状态,这些状态收集时开销以及多节点状态间的同步是多节点运行时故障检测需要解决的重要问题.

PDA(Passive Distributed Assertion)通过引入被动的分布式断言对多节点交互故障进行检测[9].PDA在传统断言中引入节点集合操作符,允许用户在同一时刻为涉及多个空间上分布的节点状态创建断言,并在系统运行过程中使用网络监听方式收集相关节点状态信息,然后再集中地进行断言的检查.为了确保断言检查结果正确性,PDA需要消除监听记录中因数据丢失和时间同步精度导致的误差.然而,断言失效并不总意味着系统存在故障,有效的断言依赖于用户对系统状态正确的假设.当涉及多个节点状态时,正确的断言有时并不像传统程序中那样容易建立.由于断言涉及到多个节点内部程序状态,PDA最主要的开销来自这些状态的收集.与PDA类似,Lodder等人[72]提出利用传感器网络运行记录以离线的方式重构网络的全局状态,供用户对其一致性进行检查与分析;EvAnT[73]则从事件分析角度对网络中节点状态建模,允许用户通过创建事件查询的方式对涉及多个节点的故障进行检测.

(3)基于形式化和其他静态分析的故障检测

传统测试方法允许开发人员得到一组特定参数配置下的执行结果,但大量参数和每种参数可能值

产生的组合爆炸使其无法揭示所有情况下的故障.由于节点软件规模较小并且任务调度执行方式相对简单,这使得应用形式化和其他静态分析技术对软件进行故障检测成为可能.例如,有研究人员通过模型检验(model checking)[74]和符号执行(symbolic execution)[75]等方法对传感器网络系统状态空间进行搜索,揭示一部分因非确定性事件导致的故障.然而,随着网络中节点数和拓扑复杂度的增加,这类方法会遇到状态空间爆炸问题.程序代码到检验模型的自动转换以及状态空间的有效化简是这类故障检测方法能否有效应用的关键.目前,这一方面问题开始得到软件工程领域研究人员的关注.结合传感器网络程序特点,将现有软件分析技术中相关成果[14]进行应用和改进,是这一领域未来需要探索的内容.

4.1.4　其它故障检测技术

能耗是传感器网络一项重要的指标参数,为了确保部署系统可以长时间运行,传感器节点大部分时间处于低功耗休眠状态,因此节点运行过程中能耗异常可以用于系统故障检测与分析.节点能耗计量可分为硬件计量和软件计量两种方式.硬件计量需要修改传感器节点硬件,其附加成本随测量精度提高而增加[76-77].软件计量没有附加成本且精度适中,通过对节点能耗的预先测量与校准,再结合节点硬件在不同能耗状态下的运行时间统计结果,可以对能耗进行较为准确的估计[78].此外,借助软件计量用户还可获得程序不同模块的能耗使用情况,并使用这些信息对系统进行更加深入的分析[79].文献[78-80]借助能耗测量对节点程序及网络性能进行了分析.Khan 等人[47]利用节点能耗建模对路由失效、天线失效和系统崩溃等典型故障进行检测.

4.2　故障定位

故障修复的前提是准确地隔离故障,查找导致故障的软件缺陷位置,即故障定位.

4.2.1　可视化与交互技术

传感器节点的人机接口通常十分有限,一般仅配备有限的 LED 灯和按键,这为节点的内部状态检查,特别是部署时和部署后的现场调试带来了诸多不便.SeeDTV[81]提供的带有图形化接口的手持设备,可以让用户迅速对部署现场中网络无线通信数据、节点硬件接口、传感器 ADC 读数以及电量消耗等信息实施检查,从而帮助用户更加快速有效地对故障进行分析.在实际应用部署中,SeeDTV 帮助实验人员发现了两个通信功能异常节点,一个能耗异常节点以及一个 ADC 异常节点.此外 Ganju 等人[82]研究了基于声信号反馈的辅助调试技术;Gauger 等人[83]则对手势、指向和扫描三种与传感器节点的交互方式进行了研究,这些技术适用于某些特殊情况下的调试.

除了部署环境中与节点交互上的限制,缺乏统一、通用的人机交互接口使传感器网络故障定位变得繁琐而又容易出错.网络拓扑、节点状态以及感知数据的可视化可以帮助用户更加有效的对系统状态进行分析.Mote-View[45]是一款商业化的传感器网络可视化监控与管理工具,但缺乏通用性与可扩展性.Octopus[84]是一款与 Mote-View 功能类似开放源代码的传感器网络可视化与管理工具,其平台无关性以及丰富的配置接口使其可以方便地同现有节点软件平台(如 TinyOS 和 Contiki 等)进行集成.然而,当用户需要对数据进行一些相对复杂的处理时,上面列举的可视化工具无法很好地满足这类用户的需求.针对这一问题,tinyLAB[85]基于 Matlab 计算平台向用户提供了传感器网络数据实时分析、可视化与交互环境.此外,分析网络性能时,无线监听器获得的大量数据包往往需要通过可视化技术帮助用户更加直观的对网络的各项性能进行分析.现有针对传感器网络无线通信标准的商业化网络分析工具如 Perytons Analyzer① 等都提供了这样的功能.

4.2.2　源码调试器

源码调试器(source-level debugger)允许用户在程序中任意位置设置断点,并在断点激活后对程序状态进行检查与修改,是一种常用的故障定位与分析工具.借助编译器生成的源码/机器码符号映射表,源码调试器——有时又被称作符号调试器(symbolic debugger)或者被简称为调试器——向用户提供了源代码级别的调试视图.

源码调试器常见的实现方式有 4 种.大多数调试器借助处理器内置特殊寄存器实现断点以及监测变量地址的存储,当程序计数器或内存访问地址与这些寄存器中的内容相匹配时,处理器会通过特殊的调用将控制权交给调试器.然而由于大多数传感器节点使用的处理器通常不包含这样的功能,因此该方法不适用于传感器网络.第 2 种方法通过程序运行时系统提供的控制接口(如 Java Debug API)实现,然而由于资源的限制,大多数传感器网络程序直接在硬件而非某种运行时系统上执行,因此该方法也不具有普遍性.第 3 种方法通过 In-Circuit

① Perytons Analyzer. http://www.perytons.com

Emulator(ICE)硬件仿真实现,调试器借助 ICE 通过 JTAG(IEEE 1149.1)接口实现对节点处理器的控制以及内部状态的提取.由于该方法需要每个被调试节点与一个单独的 ICE 物理相连,因此并不适用于部署后或由过多节点构成的传感器网络.第 4 种方法通过目标码动态插装实现程序断点和控制权的转移,该方法不用修改源代码且无需额外硬件的支持,因此可用于部署后传感器网络的远程调试.

Clairvoyant[27]通过目标码动态插装为 TinyOS 实现了一个类 GNU GDB 的远程调试器,其节点上额外软件的程序存储(ROM)和数据存储(RAM)开销分别为 32 KB 和 1 KB. Clairvoyant 使用 Trickle[86]协议和专门消息分配器,解决传感器网络多跳环境下 PC 与节点之间相关调试信息的传输问题,然而 Clairvoyant 与应用程序必须采用相似的底层通信协议栈.此外,Clairvoyant 调试信息采用的带内传输方式会对应用本身的通信造成一定影响.最后由于需要对程序 ROM 存储器的内容进行改写,调试器的使用会受存储器擦写次数以及擦写电压的限制.与基于源码插装的故障定位技术相比,Clairvoyant 允许用户在无需修改源码的情况下,以交互的方式对程序的执行以及状态进行检查,在某些情况下可以大大简化故障定位与分析的时间.然而受网络通信方式、程序存储空间、存储器擦写以及调试器探针效应等方面的限制,Clairvoyant 并不适用于故障定位的所有情况.

远程源代码调试器允许用户以控制执行和检查程序状态的方式对程序执行进行深入的分析,但其实现方式较为复杂.在某些情况下简单的状态查询以及远程过程调用同样可帮助调试人员实施故障的定位与分析. SNMS[32]借助程序编译时产生的符号表信息允许用户查询和修改 TinyOS 程序组件的变量状态. Marionette[61]则在其基础之上为 TinyOS 添加了嵌入式远程过程调用(Embedded RPC)支持.此外,LiteOS 和 Contiki 提供的远程 Shell 功能允许用户对节点实施一些简单的远程控制和检查.

4.2.3 程序执行记录

调试器允许用户在不添加任何额外代码的情况下对程序的执行进行控制与检查,然而在合适的位置设置断点检查程序状态,往往需要用户对故障具有一定的认识.当断点难以设置或者无法使用时,用户需要借助程序执行记录对故障进行事后分析和定位.传感器节点有限的资源与其运行过程中产生的大量状态信息(如传感器读数、程序中变量的修改、函数的调用以及代码执行路径等)为程序执行记录提出了挑战.

(1) AOP(Aspect Oriented Programming)技术

程序执行记录一般通过代码插装实现.与传统日志使用方式相比,节点有限的资源以及潜在故障的不可预知性,使得用户需要一种更加便捷的手段,根据实际运行需要设置插装代码. AOP 技术可以有效地分离系统业务逻辑与监测逻辑,使用户以简洁的方式对代码插装进行描述.例如为了监测系统某一变量的状态,用户需要在变量修改的相关位置安放插装代码,在非 AOP 的手工方式下用户可能会遗漏某些安置点,而 AOP 则向用户提供了基于模式匹配的插装描述方式,允许指定在某变量每次修改后实施插装[10,28,87]. AOP 技术可以让用户更加高效地对程序实施插装,减少了插装过程中因人工介入而导致的错误.例如,Cao 等人[28]采用 AOP 和动态目标码插装技术,允许用户在程序运行中根据需要动态地设置程序执行记录点(trace point).用户使用简洁的声明式语言在程序特定语句、函数调用入口或返回处添加记录点,其语言形式如下:

TRACE {…} FROM {…}

EXECUTE {…} WHERE {…}.

其中 TRACE 和 FROM 关键字用于限定记录点的位置,如某一文件中的某一函数,用户可以使用正则表达式对文件以及函数进行筛选;WHERE 用于限定记录点生效的条件;而 EXECUTE 用于指定记录点生效时需要执行的一系列操作如调用的函数和变量的读写等.

(2)事件日志、函数调用记录以及控制流追踪

事件日志(event logging)、函数调用记录(call tracing)以及控制流追踪(control-flow tracing)从不同的粒度上反应了程序的执行情况,用户可以利用这些记录对故障进行事后的定位与分析.

虽然大部分传感器网络操作系统提供了事件日志系统,但节点有限的存储、带宽以及能量使得传统事件日志方式并不能很好的适用于传感器网络的故障分析. SNMS[32]对日志中字符常量用标识符进行再编码,减少了事件日志中不必要的存储,但仍然无法解决大量事件带来的存储问题. NodeMD[88]将事件记录限定为 15 个常见的系统事件,并结合系统中不同线程执行时对应的事件序列签名,重构出用于故障分析的运行记录.其相关的系统事件包括函数的调用与返回,线程的相关操作如创建、退出、阻塞、

休眠等，软件定时器的设置和超时以及中断触发. 虽然每个事件仅需要 4 比特的存储空间，但真实应用中产生的事件日志仍远大于节点自身的存储能力.

与随意添加的事件日志相比，程序执行时特定信息记录，如函数调用和控制流，同样可以为故障分析提供帮助. 函数调用记录可以用于故障的初步定位，而程序控制流信息则可以为故障定位提供更加详细执行分析记录. 利用程序自身结构的特征，这些信息可以被进一步有效压缩. 借助局部调用日志技术(local call logging)[89]，LIS[87]可以有效的减少传统基于全局标识符生成的函数调用记录中的冗余，用户通过指定关注区域(Region Of Interest，ROI)获取可能产生故障模块的函数调用记录. LIS 首先根据 ROI 对程序中的函数进行分类，然后根据分类进行全局标识符和局部标识符的分配. LIS 通过在拥有全局标识符的函数中使用局部标识符记录函数调用，将原先全局的名字空间分解为多个较小局部名字空间，进而有效地减少记录中标识符的平均比特宽度. 实验表明随着需要记录函数个数的增加，局部调用日志技术可以有效的将记录大小减少到使用全局标识符时的 40％到 65％. Sundaram 等人[90]认为程序执行时产生的控制流记录可以帮助用户解决大部分情况下的故障分析问题，并基于 BL 路径编码[91]算法为 TinyOS 出了一种高效的控制流追踪技术. 该技术通过记录系统运行过程中事件、任务以及这些执行单元内部控制流，并利用程序执行的周期性特征对记录进行压缩，从而大大减少了记录所需的存储空间.

4.2.4　重放和检查点

某些传感器网络应用的行为，如目标追踪应用，主要受感知的外部事件影响，而动态部署环境中感知事件的异步与不可重复特性，为这类应用的故障定位以及性能分析带来巨大的挑战. 重放技术很早就被用于分布式系统调试，一般通过对非确定事件如中断、I/O 以及消息的记录实现. 通过对系统故障进行重放(replay)，用户可以调整相关参数并对程序的执行进行分析，确定导致系统故障的原因. EnviroLog[92]向用户提供了异步感知事件的记录与重放服务. 在记录阶段，日志模块为用户感兴趣的事件创建带有时间戳的记录并保存在 Flash 存储器中；在重放阶段，重放模块依据先前保存记录和时间戳进行事件的重放. 重放模型假设系统重放的执行不会对被重放的事件本身产生影响，即系统输出单向依赖于记录的异步事件. 由于传感器节点资源受限，重放技术的使用会受到外部事件产生的数据量以及重放时间同步精度的限制.

与重放技术类似，检查点(checkpointing)技术通过在程序执行过程中周期性创建检查点，使用户可以利用故障发生前后附近的检查点将系统恢复到相应的状态，从而实现故障的再现. 由于需要存储整个系统的状态，检查点技术通常无法应用于资源受限且已部署的传感器网络. Osterlind 等人[93]为传感器网络测试台提出一了种检查点技术，该技术可以为测试台上的运行程序周期性创建检查点，并允许用户将检查点对应的系统状态转移到模拟器中，实现检查点跨平台加载. 中央服务器通过向节点发送冻结与解冻命令实现检查点的创建与恢复. 为了保证获取到的整个网络状态的一致性，需要尽可能减少网络中所有节点检查点创建的时间同步误差. 除了故障定位，该技术可以应用于如测试台的重复实验、故障注入测试以及模拟模型验证等很多领域.

4.2.5　其它故障定位技术

(1) 基于数据挖掘的日志分析

运行环境的不可控性使得传感器网络中的有些故障往往难以再现，如协议设计中缺陷或多组件集成导致的组件之间的交互问题. 用户需要借助事后分析手段对这类故障进行定位，但节点资源上的限制使得用户无法获得足够的事后调试信息. 应用数据挖掘算法对故障日志进行分析，将有效地帮助用户对故障成因进行推断. Dustminer[94]通过收集系统运行过程中正常与故障两种情况下的事件日志，利用关联规则挖掘算法(Apriori)提取两种情况下判别模式(discriminative patters)，即两种情况下特有的事件序列. 用户通过对比与分析这些特有序列对故障原因做出推断. 类似的，SNTS[95]使用 PART 算法[96]得到故障与正常两种情况下日志记录对应的判别规则，用户通过分析这些规则对故障成因进行推断.

(2) 编程抽象对故障定位的支持

全新的应用需求以及受限的资源为传感器网络应用编程提出了挑战，人们为传感器网络设计了很多新颖的编程抽象，以简化编程、帮助用户快速地构建应用. 然而，当用户构建的应用出现问题，并且相应编程抽象又缺乏对调试的支持时，编程抽象的优势将不复存在，用户需要分析生成代码对相关故障进行定位. 因此，编程抽象在设计时，需要考虑向用户提供故障定位方面的支持[97]. YETI[98]利用 TinyOS 的 nesC 编译器产生的 nesC 语言到 C 语言

的映射信息向用户提供了支持 nesC 编程抽象调试的集成开发环境. MDB[99] 则允许用户以顺序的方式对 MacroLab[100] 宏编程代码进行调试,而这些代码对应的节点程序实际是以异步、分布式的方式在多个节点上执行的.

4.3 故障修复

故障修复是调试过程中的重要一环,其目的是对部署后的传感器网络系统及应用进行缺陷排除、功能升级和性能优化以适应周边物理环境和自身动态系统的变化. 与传统故障修复相比,传感器网络故障修复除了遵循通用的原因分析、缺陷排除、复查测试和确认正常等步骤外,还具有如下特点:

(1) 原地(in situ)故障修复

由于众多感知节点深度嵌入于物理环境之中,系统一旦部署,在许多应用场景下(如野生动物习性监测、深海、火山喷发口),物理接触全部节点并不现实,因此需要借助无线链路对远程节点上运行的软件缺陷进行修复. 实现该过程的途径通常被称为"空中"重编程(over-the-air reprogramming).

(2) 故障修复能力与故障修复代价存在矛盾

传感器网络故障修复能力取决于空中重编程的粒度和节点执行环境能力. 因为不同层次的抽象级别使得对底层的完整执行程序或程序部分模块更新比对高层的虚拟机脚本更新具有更强、更灵活的修复能力. 而现有的研究已表明传感器节点传输信息要比执行计算更消耗能量,这使得想要获得更高的故障修复能力就可能产生更大的故障修复代价.

4.3.1 代码更新

代码更新是传感器网络软件故障修复的主要手段. 作为传感器网络中一个重要研究领域,人们对代码更新进行了深入的研究. 本文仅对代码更新技术进行简要介绍,关于这方面更加深入的论述,读者可以参考代表性的文献和相关综述[101-103]. 根据不同的故障修复能力,我们将传感器网络代码更新方法分为 3 类:全映像更新、模块级更新、虚拟机级更新.

(1) 全映像更新

有些节点操作系统(如 TinyOS[20] 和 MOS[104])通过在编译时静态优化代码的方法,达到对处理器和存储资源的合理使用,以提高系统的运行效率. 在这样的执行环境中,操作系统和应用程序混合编译成一个单一的可执行代码,因此不管是应用程序还是操作系统在修复时,都需要对整个程序映像进行更新. 这种情况下代码传输代价会很大,给故障修复带来巨大挑战. 这类更新技术研究的重点是如何降低大块数据分发时的网络传输开销,并尽可能减少整个网络代码更新时间. 其中,具有代表性的研究有 Deluge[101]、MNP[105] 和 Rateless Deluge[106] 等. 增量式更新是对全映像更新的一种改进[101],它假设每次更新只修改了少量的代码. 通过只传输更新映像与原映像的不同部分,增量更新可以有效的减少代码的传输量.

(2) 模块级更新

这类更新技术一般需要节点执行环境具有动态模块加载功能(如 SOS[29] 和 Contiki[22]). 由于模块之间是一种松散的耦合,所以模块之间的调用比简单的函数调用需要更大的开销,同时由于缺乏编译时的全局优化,因此在代码体积和执行效率上,比单一化执行环境要低. 但是模块化执行环境具有动态的模块加载能力,使得在传感器网络故障修复时,只需更新部分的程序模块,而不用将整个程序的代码全部更新. 模块级修复在传输和更新代价上比全映像更新要小,因此可以节省更多的能量,延长传感器网络的生命周期.

(3) 虚拟机级更新

如果故障修复涉及的代码执行环境是虚拟机时(如 Mate[26] 和 ASVM[25]),由于传输的是虚拟机脚本,其更新代码量远低于二进制映像代码量. 因此在虚拟机执行环境中,更新修复在网络中的传输开销比前两种执行环境都要小. 此外,虚拟机解释执行的特点,使程序在执行时不能直接访问硬件,而只能间接地通过虚拟机完成,因此虚拟级的故障修复具有良好的执行安全性. 但是中间代码解释执行的效率比前两种执行环境都低. 另外,如何选定适合的虚拟机指令集是一个挑战性的问题. ASVM 提供了一种静态的应用相关的虚拟机生成框架,它可以针对应用的特点来定义相应的虚拟机执行环境. DAViM[107] 和 DVM[108] 都试图提供一种支持可动态重配置的通用虚拟机执行环境来进一步提高故障修复的能力.

表 5 不同代码更新方式的比较

	全映像更新	增量式更新	模块级更新	虚拟机级更新
修改能力	高	高	中	低
更新开销	高	依情况	中	低
是否重启	是	是	否	否

最后,表 5 对不同代码更新方式在修改能力、更新开销以及更新后节点是否需要重启等方面的特点进行了总结与比较.

4.3.2 软件抗衰技术(software rejuvenation)

与物理世界紧密耦合以及长期运行的特性,使传感器网络会受到"软件衰退"的影响[13,70]. 软件衰

退是指一个长时间持续运行的软件系统会发生状态退化和性能降低，最终导致系统崩溃[12]. 软件衰退的根本原因是系统中的软件缺陷，这类缺陷往往因错误传播模型复杂而难以进行故障定位，当系统资源高度受限时故障定位将变得异常困难.

然而，在相关缺陷被彻底从系统中移除之前，我们可以借助软件抗衰技术对相关故障进行修复. 软件抗衰技术的基本思想是通过周期性地暂停软件的运行，清除持续运行系统的内部状态、重新启动并恢复为初始状态或“健康的”中间状态，预防将来可能发生的更严重的故障[109]. 节点重启是传感器网络中最简单也是最常用的一种抗衰方法，但这种方式的代价较高. 因为简单重启会影响保存在数据存储器中节点所有的软件状态，其中的某些状态重建需要消耗节点的能源，如路由信息、邻居表等状态信息. 当软件衰退部分与这些状态无关时，整个系统的重启会导致过多的开销. Chen 等人[70]研究了节点内存错误处理时的微重启技术，而 Woehrle 等人[13]则从组件角度研究了细粒度重启的节点软件抗衰技术. 这些研究表明软件抗衰技术在传感器网络中应用的可行性与有效性，是传感器网络一个重要的研究方向. 未来传感器网络软件抗衰技术需要在系统结构设计、恢复时机以及不同恢复策略对系统开销的影响等方面进行深入的研究.

5　未来研究方向

随着越来越多实际应用系统的出现，越来越多的研究人员开始关注传感器网络调试以及与实际部署相关的各类问题，同时提出了不少新颖的解决方案和技术[6-7]. 但目前这个领域的研究还处于一个起步阶段，我们认为未来的热点研究方向包括以下一些内容.

5.1　部署前测试与验证

通过各种测试与验证手段尽可能在部署前发现系统潜在缺陷，可以在一定程度上减少在部署后资源受限条件下实施调试可能. 现有的模拟仿真工具通常对感知和无线传输等模型做了过多简化和假设，并且大多注重网络通信层面的性能分析，无法对一些感知数据驱动的应用——如目标追踪和危险气体扩散检测等——进行有效的模拟分析. 未来需要研究典型应用场景建模与仿真技术，通过完善、优化无线传输和感知数据模型，允许用户在系统实际部署前借助模拟器对系统的功能与性能进行较为全面的评估. 此外，未来基于测试台实验结果的模拟增强技术、非确定性故障注入的测试技术以及支持复杂环境通信干扰模拟的测试台技术研究，将有效提高模拟与测试环境的真实度，帮助开发人员尽早捕获系统潜在缺陷. 此外，已有研究表明静态软件分析技术可以有效地揭示某些与非确定性故障相关软件缺陷. 结合节点程序特性以及网络约束关系，将已有静态软件分析技术应用于传感器网络故障检测，也是未来部署前调试技术需要深入研究的内容.

5.2　操作系统和编程抽象对调试的支持

为了满足传感器网络全新的应用需求，新颖的操作系统和编程抽象一直是人们研究的一个热点，然而这些操作系统和编程抽象在设计时很少考虑如何为构建的应用提供专门的调试支持[97]. 未来的研究，人们一方面需要为现有操作系统和编程抽象增加调试支持，如本文 4.2.5 节提到的 MDB 调试器；另一方面则需要从调试角度出发，重新考虑操作系统、运行环境以及编程抽象的设计，从而为构建的应用提供有效的调试手段和验证机制. 例如有研究人员提出通过 RESTful 接口抽象出节点硬件的基本功能，将应用逻辑从节点程序中剥离[11]，开发人员通过这些接口在节点嵌入式软件之上构建应用，进而缓解了传感器网络调试时因受限资源导致的各类问题. 这类思想也被用于解决普适计算环境中异构设备集成时的可扩展性问题[110].

5.3　长期监测与事后(post-mortem)调试

持续运行以及资源受限特性，向传感器网络的长期监测提出了挑战. 调试需要系统状态可见，然而可见度和资源开销之间存在矛盾. 传感器网络监测需要在足够的可见度和有限的资源之间寻求一个合适的平衡点. 特别是当需要对系统进行长时间观测以收集足够的信息对相关问题进行推断时，这种平衡显得更加的重要. 例如，vLevels[10]为系统监测提出的可见度管理机制以及 ADA[111]提出的自适应调试方法，都考虑了调试中资源开销与状态信息的平衡问题. 此外，低功耗、大容量以及低成本 Flash 存储设备的出现，使得用户可以在节点本地记录更多的系统运行情况供后期故障定位使用. 未来以存储为中心的事后调试技术研究将有效克服现有集中式在资源开销大以及状态信息不足方面的缺点.

5.4　异构环境下故障定位与分析

随着智能设备的增加以及物联网的兴起，越来越多的传感器网络应用将会涉及多种计算、通信和

感知能力各异的设备.这些设备在软硬件以及协议标准上的差异会为传感器网络调试带来更大的挑战.这也是以后物联网在应用集成过程中需要考虑和解决的一个重要问题.

6 结束语

随着实际应用系统的部署,传感器网络调试问题得到了越来越多的关注.本文全面分析了传感器网络调试所涉及的问题与面临的挑战,详细总结了调试过程中常用的系统状态信息及其获取技术,综述了调试中具有代表性的故障检测、定位与修复技术,并在最后指出了未来的研究方向.我们相信,随着传感器网络理论与技术研究的深入,越来越多基于无线传感技术的应用将被布署在生产和生活的各个领域.

致 谢 审稿人为本文提出了宝贵的修改意见,作者在此表示衷心感谢!

参 考 文 献

[1] Wu He-Quan. Review on Internet of Things: Application and challenges. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition), 2010, 22(5): 526-531(in Chinese)
(邬贺铨. 物联网的应用与挑战综述. 重庆邮电大学学报(自然科学版). 2010, 22(5): 526-531)

[2] Werner-Allen G, Lorincz K, Johnson J et al. Fidelity and yield in a volcano monitoring sensor network//Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI'06). Seattle, USA, 2006: 381-396

[3] Barrenetxea G, Ingelrest F, Schaefer G et al. The hitchhiker's guide to successful wireless sensor network deployments//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 43-56

[4] Mo L, He Y, Liu Y et al. Canopy closure estimates with GreenOrbs: Sustainable sensing in the forest//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 99-112

[5] Ma J, Zhou X, Li S et al. Connecting agriculture to the Internet of Things through sensor networks//Proceedings of the 2011 IEEE International Conferences on Internet of Things, and Cyber, Physical and Social Computing (IEEE iThings/CPSCom). Dalian, China, 2011: 184-187

[6] Beutel J, Römer K, Ringwald M et al. Deployment techniques for sensor networks//Ferrari G. Sensor networks: Signals and Communication Technology. Heidelberg: Springer, 2009: 219-248

[7] Schoofs A, O'Hare G, Ruzzelli A. Debugging low-power and lossy wireless networks: A survey. IEEE Communications Surveys & Tutorials, 2012, 99(pp): 1-11

[8] Ringwald M, Romer K, Vitaletti A. Passive inspection of sensor networks//Proceedings of the 3rd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'07). Santa Fe, USA, 2007: 205-222

[9] Romer K, Ma J. PDA: Passive distributed assertions for sensor networks//Proceedings of the 2009 International Conference on Information Processing in Sensor Networks (IPSN'09). San Francisco, USA, 2009: 337-348

[10] Ma J, Römer K. Visibility levels: Managing the tradeoff between visibility and resource consumption//Proceedings of the 4th International Conference on Real-World Wireless Sensor Networks (REALWSN'10). Colombo, Sri Lanka, 2010: 49-61

[11] Kovatsch M. Firm firmware and apps for the internet of things//Proceedings of the 2nd Workshop on Software Engineering for Sensor Network Applications (SESENA'11). Waikiki, USA, 2011: 61-62

[12] Grottke M, Trivedi K S. Fighting bugs: Remove, retry, replicate, and rejuvenate. IEEE Computer, 2007, 40(2): 107-109

[13] Woehrle M, Meier A, Langendoen K. On the potential of software rejuvenation for long-running sensor network deployments//Proceedings of the 2010 ICSE Workshop on Software Engineering for Sensor Network Applications (SESENA'10). Cape Town, South Africa, 2010: 44-48

[14] Mei Hong, Wang Qian-Xiang, Zhang Lu et al. Software analysis: A road map. Chinese Journal of Computers, 2009, 32(9): 1697-1710(in Chinese)
(梅宏, 王千祥, 张路等. 软件分析技术进展. 计算机学报, 2009, 32(9): 1697-1710)

[15] Ramanathan N, Schoellhammer T, Kohler E et al. Suelo: Human-assisted sensing for exploratory soil monitoring studies//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 197-210

[16] Chen B, Peterson G, Mainland G et al. LiveNet: Using passive monitoring to reconstruct sensor network dynamics//Proceedings of the 4th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'08). Santorini Island, Greece, 2008: 79-98

[17] Brown J, Mccarthy B, Roedig U et al. BurstProbe: Debugging time-critical data delivery in wireless sensor networks//Proceedings of the 8th European Conference on Wireless Sensor Networks (EWSN'11). Bonn, Germany, 2011: 195-210

[18] Choi J I, Lee J W, Wachs M et al. Opening the sensornet black box. ACM SIGBED Review, 2007, 4(3): 13-18

[19] Wachs M, Choi J I, Lee J W et al. Visibility: A new metric for protocol design//Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys'07). Sydney, Australia, 2007: 73-86

[20] Hill J, Szewczyk R, Woo A et al. System architecture directions for networked sensors//Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX). Cambridge, USA, 2000: 93-104

[21] Gnawali O, Fonseca R, Jamieson K et al. Collection tree protocol//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 1-14

[22] Dunkels A, Gronvall B, Voigt T. Contiki — A lightweight and flexible operating system for tiny networked sensors//Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04). Tampa, USA, 2004: 455-462

[23] Beutel J. Fast-prototyping using the BTnode platform//Proceedings of the Conference on Design, Automation and Test in Europe (DATE'06). Munich, Germany, 2006: 977-982

[24] Cao Q, Abdelzaher T, Stankovic J et al. The LiteOS operating system: Towards unix-like abstractions for wireless sensor networks//Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN'08). St. Louis, USA, 2008: 233-244

[25] Levis P, Gay D, Culler D. Active sensor networks//Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2 (NSDI'05). Boston, USA, 2005: 343-356

[26] Levis P, Culler D. Mate: A tiny virtual machine for sensor networks//Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X). San Jose, USA, 2002: 85-95

[27] Yang J, Soffa M L, Selavo L et al. Clairvoyant: A comprehensive source-level debugger for wireless sensor networks//Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys'07). Sydney, Australia, 2007: 189-203

[28] Cao Q, Abdelzaher T, Stankovic J et al. Declarative tracepoints: A programmable and application independent debugging system for wireless sensor networks//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 85-98

[29] Han C, Kumar R, Shea R et al. A dynamic operating system for sensor nodes//Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys'05). Seattle, USA, 2005: 163-176

[30] Kothari N, Nagaraja K, Raghunathan V et al. HERMES: A software architecture for visibility and control in wireless sensor network deployments//Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN'08). St. Louis, USA, 2008: 395-406

[31] Hunt G, Brubacher D. Detours: Binary interception of Win32 functions//Proceedings of the 3rd Conference on USENIX Windows NT Symposium-Volume 3 (WINSYM'99). Seattle, USA, 1999: 14

[32] Tolle G, Culler D. Design of an application-cooperative management system for wireless sensor networks//Proceedings of the Second European Workshop on Wireless Sensor Networks (EWSN'05). Istanbul, Turkey, 2005: 121-132

[33] Zhao J, Govindan R, Estrin D. Computing aggregates for monitoring wireless sensor networks//Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03). Anchorage, USA, 2003: 139-148

[34] Rost S, Balakrishnan H. Memento: A health monitoring system for sireless sensor networks//Proceedings of the 3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'06). Reston, USA, 2006: 575-584

[35] Meier A, Motani M, Siquan H et al. DiMo: Distributed node monitoring in wireless sensor networks//Proceedings of the 11th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'08). Vancouver, Canada, 2008: 117-121

[36] Liu K, Ma Q, Zhao X et al. Self-diagnosis for large scale wireless sensor networks//Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM'11). Shanghai, China, 2011: 1539-1547

[37] Liu K, Li M, Liu Y et al. Passive diagnosis for wireless sensor networks//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 113-126

[38] O'Donovan T, Tsiftes N, He Z et al. Detailed diagnosis of performance anomalies in sensornets//Proceedings of ACM HotEMNETS 2010 Workshop on Hot Topics in Embedded Networked Sensors (HotEmNets'10). Killarney, Ireland, 2010: 1-9

[39] Mathur G, Desnoyers P, Chukiu P et al. Ultra-low power data storage for sensor networks. ACM Transactions on Sensor Networks, 2009, 5(4): 31-33

[40] Tsiftes N, Dunkels A, He Z et al. Enabling large-scale storage in sensor networks with the Coffee file system//Proceedings of the 2009 International Conference on Information Processing in Sensor Networks (IPSN'09). San Francisco, USA, 2009: 349-360

[41] Tsiftes N, Dunkels A. A database in every sensor//Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys'11). Seattle, USA, 2011: 316-332

[42] Yeo J, Youssef M, Agrawala A. A framework for wireless LAN monitoring and its applications//Proceedings of the 3rd ACM Workshop on Wireless Security (WiSe'04). Philadelphia, USA, 2004: 70-79

[43] Xu X, Wan J, Zhang W et al. PMSW: A passive monitoring system in wireless sensor networks. International Journal of Network Management. 2011, 21(4): 300-325

[44] Dyer M, Beutel J, Kalt T et al. Deployment support network - a toolkit for the development of WSNs//Proceedings of the 4th European Conference on Wireless Sensor Networks (EWSN'07). Delft, Netherlands, 2007: 195-211

[45] Werner-Allen G, Swieskowski P, Welsh M. MoteLab: A wireless sensor network testbed//Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN'05). Los Angeles, USA, 2005: 483-488

[46] Huangfu W, Sun L, Zhou X. NISAT: A zero-side-effect testbed for wireless sensor networks//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 313-314

[47] Khan M M H, Le H K, Lemay M et al. Diagnostic power-tracing for sensor node failure analysis//Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'10). Stockholm, Sweden, 2010: 117-128

[48] Ni K, Ramanathan N, Chehade M N H A et al. Sensor network data fault types. ACM Transactions on Sensor Networks, 2009, 5(3): 21-25

[49] Gao J, Xu Y, Li W. Weighted-median based distributed fault detection for wireless sensor networks. Journal of Software, 2007, 18(5): 1208-1217

[50] Rajagopal R, Nguyen X, Ergen S C et al. Distributed online simultaneous fault detection for multiple sensors//Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN'08). St. Louis, USA, 2008: 133-144

[51] Sharma A B, Golubchik L, Govindan R. Sensor faults: Detection methods and prevalence in real-world datasets. ACM Transactions on Sensor Networks, 2010, 6(3): 21-23

[52] Guo S, Zhong Z, He T. FIND: Faulty node detection for wireless sensor networks//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 253-266

[53] Ramanathan N, Chang K, Kapur R et al. Sympathy for the sensor network debugger//Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SenSys'05). San Diego, USA, 2005: 255-267

[54] Miao X, Liu K, He Y et al. Agnostic diagnosis: Discovering silent failures in wireless sensor networks//Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM'11). Shanghai, China, 2011: 1548-1556

[55] Awad A, Nebel R, German R et al. On the need for passive monitoring in sensor networks//Proceedings of the 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools (DSD'08). Parma, Italy, 2008: 693-699

[56] Girod L, Ramanathan N, Elson J et al. Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks. ACM Transactions on Sensor Networks, 2007, 3(3): 11-13

[57] Levis P, Lee N, Welsh M et al. TOSSIM: Accurate and scalable simulation of entire TinyOS applications//Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys'03). Los Angeles, USA, 2003: 126-137

[58] Titzer B L, Lee D K, Palsberg J. Avrora: Scalable sensor network simulation with precise timing//Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN'05). Piscataway, USA, 2005: 477-482

[59] Eriksson J, O Sterlind F, Finne N et al. COOJA/MSPSim: Interoperability testing for wireless sensor networks//Proceedings of the 2nd International Conference on Simulation Tools and Techniques (Simutools'09). Rome, Italy, 2009: 21-27

[60] Okola M, Whitehouse K. Unit testing for wireless sensor networks//Proceedings of the 2010 ICSE Workshop on Software Engineering for Sensor Network Applications (SESENA'10). Cape Town, South Africa, 2010: 38-43

[61] Whitehouse K, Tolle G, Taneja J et al. Marionette: Using RPC for interactive development and debugging of wireless embedded networks//Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN'06). Nashville, USA, 2006: 416-423

[62] Veress K, Maroti M. LinkBench: Benchmark and metric framework for wireless sensor networks//Proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN'11). Chicago, USA, 2011: 171-172

[63] He Z, Voigt T. Precise packet loss pattern generation by intentional interference//Proceedings of the 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS'11). Barcelona, Spain, 2011: 1-6

[64] Boano C A, Voigt T, Noda C et al. JamLab: Augmenting sensornet testbeds with realistic and controlled interference generation//Proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN'11). Chicago, USA, 2011: 175-186

[65] Xiong J, Ngai E C H, Zhou Y et al. RealProct: Reliable protocol conformance testing with real nodes for wireless sensor networks//Proceedings of the 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'11). Changsha, China, 2011: 572-581

[66] Shore J. Fail fast. IEEE Software, 2004, 21(5): 21-25

[67] Gay D, Levis P, von Behren R et al. The nesC language: A holistic approach to networked embedded systems//Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation (PLDI'03). San Diego, USA, 2003: 1-11

[68] Cooprider N, Archer W, Eide E et al. Efficient memory safety for TinyOS//Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys'07). Sydney, Australia, 2007: 205-218

[69] Zhou F, Condit J, Anderson Z et al. SafeDrive: Safe and recoverable extensions using language-based techniques//Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI'06). Berkeley, USA, 2006: 45-60

[70] Chen Y, Gnawali O, Kazandjieva M et al. Surviving sensor network software faults//Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles (SOSP'09). Big Sky, USA, 2009: 235-246

[71] Archer W, Levis P, Regehr J. Interface contracts for TinyOS//Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN'07). Cambridge, USA, 2007: 158-165

[72] Lodder M, Halkes G P, Langendoen K G. A global-state perspective on sensor network debugging//Proceedings of the 5th ACM Workshop on Hot Topics in Embedded Networked Sensors (HotEmNets'08). Charlottesville, USA, 2008: 37-41

[73] Woehrle M, Plessl C, Lim R et al. EvAnT: Analysis and checking of event traces for wireless sensor networks//Proceedings of the 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'08). Taichung, China, 2008: 201-208

[74] Li P, Regehr J. T-check: Bug finding for sensor networks//Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'10). Stockholm, Sweden, 2010: 174-185

[75] Sasnauskas R, Landsiedel O, Alizai M H A W et al. KleeNet: Discovering insidious interaction bugs in wireless sensor networks before deployment//Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'10). Stockholm, Sweden, 2010: 186-196

[76] Dutta P, Feldmeier M, Paradiso J et al. Energy metering for free: Augmenting switching regulators for real-time monitoring//Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN'08). St. Louis, USA, 2008: 283-294

[77] Jiang X, Dutta P, Culler D et al. Micro power meter for energy monitoring of wireless sensor networks at scale//Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN'07). Cambridge, USA, 2007: 186-195

[78] Dunkels A, Osterlind F, Tsiftes N et al. Software-based online energy estimation for sensor nodes//Proceedings of the 4th Workshop on Embedded Networked Sensors (Emnets'07). Cork, Ireland, 2007: 28-32

[79] Fonseca R, Dutta P, Levis P et al. Quanto: Tracking energy in networked embedded systems//Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI'08). San Diego, USA, 2008: 323-338

[80] Martins M, Fonseca R, Schmid T et al. Network-wide energy profiling of CTP//Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10). Zurich, Switzerland, 2010: 439-440

[81] Liu H, Selavo L, Stankovic J. SeeDTV: Deployment-time validation for wireless sensor networks//Proceedings of the 4th Workshop on Embedded Networked Sensors (EmNets'07). Cork, Ireland, 2007: 23-27

[82] Ganju D, Schwiebert L. Using sound for monitoring wireless sensor network behavior//Proceedings of the 2007 ACM/IFIP/USENIX International Conference on Middleware Companion (MC'07). Newport Beach, USA, 2007: 1-5

[83] Gauger M, Saukh O, Marron P J. Talk to me! on interacting with wireless sensor nodes//Proceedings of the 2009 IEEE International Conference on Pervasive Computing and Communications (PerCom'09). Galveston, USA, 2009: 1-8

[84] Jurdak R, Ruzzelli A, Baribirato A et al. Octopus: Monitoring, visualization, and control of sensor networks. Wireless Communication and Mobile Computing, 2011, 11(8): 1073-1091

[85] Santini S, Ostermaier B, Vitaletti A. First experiences using wireless sensor networks for noise pollution monitoring//Proceedings of the Workshop on Real-World Wireless Sensor Networks (REALWSN'08). Glasgow, Scotland, 2008: 61-65

[86] Levis P, Patel N, Culler D et al. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks//Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation (Volume 1). Berkeley, USA, 2004: 2

[87] Shea R, Srivastava M, Cho Y. Scoped identifiers for efficient bit aligned logging//Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE'10). Dresden, Germany, 2010: 1450-1455

[88] Krunic V, Trumpler E, Han R. NodeMD: Diagnosing node-level faults in remote wireless sensor systems//Proceedings of the 5th International Conference on Mobile Systems, Applications and Services (MobiSys'07). San Juan, Puerto Rico, 2007: 43-56

[89] Shea R, Srivastava M B, Cho Y. Optimizing bandwidth of call traces for wireless embedded systems. IEEE Embedded Systems Letters, 2009, 1(1): 28-32

[90] Sundaram V, Eugster P, Zhang X. Efficient diagnostic tracing for wireless sensor networks//Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10). Zurich, Switzerland, 2010: 169-182

[91] Ball T, Larus J R. Efficient path profiling//Proceedings of the 29th Annual ACM/IEEE International Symposium on Microarchitecture (MICRO 29). Paris, France, 1996: 46-57

[92] Luo L, He T, Zhou G et al. Achieving repeatability of asynchronous events in wireless sensor networks with EnviroLog//Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM'06). Barcelona, Spain, 2006: 1-14

[93] Osterlind F, Dunkels A, Voigt T et al. Sensornet checkpointing: Enabling repeatability in testbeds and realism in simulations//Proceedings of the 6th European Conference on Wireless Sensor Networks (EWSN'09). Cork, Ireland, 2009: 343-357

[94] Khan M M H, Le H K, Ahmadi H A A T et al. Dustminer: Troubleshooting interactive complexity bugs in sensor networks//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 99-112

[95] Khan M M H, Luo L, Huang C et al. SNTS: Sensor network troubleshooting suite//Proceedings of the 3rd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'07). Santa Fe, USA, 2007: 142-157

[96] Frank E, Witten I H. Generating accurate rule sets without global optimization//Proceedings of the 5th International Conference on Machine Learning (ICML'98). Madison, USA, 1998: 144-151

[97] Mottola L, Picco G P. Programming wireless sensor networks: Fundamental concepts and state of the art. ACM Computing Surveys, 2011, 43(3): 11-19

[98] Burri N, Flury R, Nellen S et al. YETI: An eclipse plug-in for TinyOS 2.1//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems(SenSys'09). Berkeley, USA, 2009: 295-296

[99] Sookoor T, Hnat T, Hooimeijer P et al. Macrodebugging: Global views of distributed program execution//Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09). Berkeley, USA, 2009: 141-154

[100] Hnat T W, Sookoor T I, Hooimeijer P et al. MacroLab: A vector-based macroprogramming framework for cyber-physical systems//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 225-238

[101] Panta R K, Bagchi S, Midkiff S P. Efficient incremental code update for sensor networks. ACM Transactions on Sensor Networks, 2011, 7(4): 30-31

[102] Wang Q, Zhu Y, Chen L. Reprogramming wireless sensor networks: Challenges and approaches. IEEE Network, 2006, 20(3): 48-55

[103] Han C, Kumar R, Shea R et al. Sensor network software update management: A survey. International Journal of Network Management, 2005, 15(4): 283-294

[104] Bhatti S, Carlson J, Dai H et al. MANTIS OS: An embedded multithreaded operating system for wireless micro sensor platforms. ACM Journal of Mobile Networks and Applications, 2005, 10(4): 563-579

[105] Kulkarni S S, Wang L. MNP: Multihop network reprogramming service for sensor networks//Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05). Columbus, USA, 2005: 7-16

[106] Hagedorn A, Starobinski D, Trachtenberg A. Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes//Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN'08). St. Louis, USA, 2008: 457-466

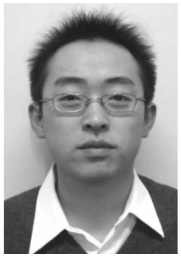
[107] Michiels S, Horr E W, Joosen W et al. DAViM: A dynamically adaptable virtual machine for sensor networks//Proceedings of the International Workshop on Middleware for Sensor Networks (MidSens'06). Melbourne, Australia, 2006: 7-12

[108] Brouwers N, Corke P, Langendoen K. A Java compatible virtual machine for wireless sensor nodes//Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08). Raleigh, USA, 2008: 369-370

[109] Xu Jian, Zhang Kun, Liu Feng-Yu. Research on software rejuvenation. Journal of Chinese Computer Systems, 2007, 28(11): 1952-1958(in Chinese)
(徐建, 张琨, 刘凤玉. 软件抗衰研究综述. 小型微型计算机系统, 2007, 28(11): 1952-1958)

[110] Chen C, Helal A S. Device integration in SODA using the device description language//Proceedings of the 9th Annual International Symposium on Applications and the Internet (SAINT'09). Seattle, USA, 2009: 100-106

[111] Li Feng, Huo Wei, Feng Xiao-Bing. An adaptive debugging approach for wireless sensor network applications. Chinese Journal of Computers, 2011, 34(7): 1195-1213(in Chinese)
(李丰, 霍玮, 冯晓兵. 面向无线传感器网络应用的自适应调试方法. 计算机学报, 2011, 34(7): 1195-1213)



MA Jun-Yan, born in 1982, Ph. D. candidate. His research interests include fault diagnosis and debugging of sensor networks.

ZHOU Xing-She, born in 1955, professor, Ph. D. supervisor. His research interests include embedded computing and pervasive computing.

ZHANG Yu, born in 1975. Ph. D., associate professor. His research interests include sensor networks and embedded operating systems.

LI Shi-Ning, born in 1967, professor, Ph. D. supervisor. His research interests include sensor networks and mobile computing.

LI Zhi-Gang, born in 1975, associate professor. His research interests include networked embedded computing and sensor networks.

Background

Sensor networks are an emerging interdisciplinary research area combining the fields of sensors, embedded computing, wireless communication and distributed processing. Various sensor networks have been deployed in the real world to help us observe our surrounding environment. However, it is common that the deployed sensor networks suffer from unpredictable faults and failures. This mainly attributes to simplified simulation models and differences between deployed and testing environment. Detecting, locating and repairing such problems are difficult due to limited access to the deployed network — both physically and due to constrained resources. As a novel computing paradigm, sensor networks pose new problems and challenges for sensor network debugging.

As increasing experimental sensor networks have been deployed in the real world, more and more attentions have been given to sensor network debugging. Debugging involves fault detecting, localization and repair. This paper provides an overview of the state of the art of debugging related techniques for sensor networks. Challenges for debugging are first addressed on the basis of analyzing the characteristics of sensor networks. System state information for debugging and its common extracting techniques are then summarized. On the basis of classification of current research results, the paper reviews principles and characteristics of representative fault detection, localization and repair techniques respectively. Finally, future research directions are proposed.

This work is supported by the National Key Technologies Research and Development Program of China (2007BAD79B00). In cooperation with Northwest Agriculture and Forestry University, the project focuses on leveraging sensor networks and the current information and communication technology to improve the production of high value-added predominant agricultural products in west China, such as apple, kiwi, melon, tomato and red sage root. Against a background of sensor network applications in agriculture, debugging techniques are investigated to enable operational and reliable sensor networks in deployed fields.