文章编号: 1001-0920 (2011) 10-1441-06

网络化控制系统瞬时故障恢复和安全控制研究综述

周纯杰[1], 黄雄峰[1,2], 秦元庆[1], 杨明月[1]

(1. 华中科技大学 控制科学与工程系，武汉 430074；2. 三峡大学 电气与新能源学院，湖北 宜昌 443002)

摘 要: 鉴于瞬时故障是导致控制系统事故的主要故障形式, 瞬时故障恢复是保证系统安全的重要手段, 首先, 介绍了当前通过主动冗余和基于系统模型分析进行瞬时故障恢复的方法; 然后, 综述这些技术在网络化控制系统的通信网络、网络节点、系统层面瞬时故障恢复和安全控制中的应用研究; 最后, 对网络化控制系统瞬时故障恢复和安全控制方法的发展趋势进行了展望.

关键词: 瞬时故障；网络化控制系统；基于模型；冗余

中图分类号: TP393 **文献标识码:** A

Survey of the transient faults recovery and safety control methods in networked control systems

ZHOU Chun-jie[1], *HUANG Xiong-feng*[1,2], *QIN Yuan-qing*[1], *YANG Ming-yue*[1]

(1. Department of Control Science and Engineering，Huazhong University of Science and Technology，Wuhan 430074，China；2. College of Electrical Engineering and Renewable Energy，China Three Gorges University, Yichang 443002，China．Correspondent：ZHOU Chun-jie，E-mail：cjiezhou@mail.hust.edu.cn)

Abstract: Transient faults are the dominant fault forms causing accidents in the control systems, and the fault recovery is a critical measure to ensure the system safety. Therefore, the typical methods of transient faults recovery and safety control based on active redundancy and system model analysis are summarized. Then the classical methods applied to recover the transient faults in communication networks, system nodes and the whole networked control systems(NCS) are surveyed. Finally, the prospect of the development of transient faults recovery and safety control in NCS is given.

Key words: transient faults；networked control systems；model-based；redundancy

1 引 言

目前, 随着计算机技术、信息技术、控制技术和网络通信技术的日益发展和交叉渗透, 基于网络通信的控制系统在航空航天、国防、电力系统、煤炭工业、汽车等众多国民经济命脉领域得到了广泛的应用. 随着控制系统规模变大、复杂程度的提高, 故障越来越频繁, 事故也不断增多. 当今社会文明程度不断提高, 人们越来越难以承受事故造成的各种损失, 使得控制系统的安全和可靠性受到人们的广泛关注[1].

故障按持续时间分为永久故障和瞬时故障[2-3]. 在现代控制系统中, 瞬时故障发生的概率远大于永久故障[4-5], 是造成事故的主要原因之一. 目前, 永久故障的恢复和安全控制已形成了系统有效的处理方法[6-7], 但瞬时故障的恢复研究还不系统, 没有完善的恢复机制. 瞬时故障造成的危害和损失可能是巨大的, 如 1990 年我国风云一号气象卫星事故和 2003 年美国大停电事故是控制系统受高能粒子影响发生瞬时故障没有得到有效控制而引发事故的典型案例[8]. 瞬时故障恢复和安全控制的系统研究, 是提高系统安全性和可靠性、减少事故发生的迫切需求.

瞬时故障恢复一般使用主动冗余和基于系统模型恢复的方法. 网络化控制系统 (NCS) 规模大, 结构复杂, 其瞬时故障具有形成原因复杂、数量多、难以定位等特点, 因此, 系统瞬时故障的恢复需要建立新的故障检测机制、多层次的恢复体系、多约束条件下的恢复策略以及形成有效的故障恢复实施网络调度方案.

收稿日期: 2011-03-12；**修回日期:** 2011-06-12.

基金项目: 国家自然科学基金项目(61074145)；高等学校博士学科点专项科研基金项目(20100142110019).

作者简介: 周纯杰(1965−), 男, 教授, 博士生导师, 从事网络化控制系统、运动控制与自动化装置等研究；黄雄峰(1980−), 男, 博士生, 从事网络化控制系统、工业通信与智能系统的研究.

2 瞬时故障恢复和安全控制方法

引起瞬时故障的原因较为复杂, 如宇宙射线的中子和粒子等瞬间改变部件 (SRAM 单元、门电路、锁存器等) 的带电状态引发存储指令或数据变更[9]、软件运行过程中的非预期交叉输入和程序 Bugs[10]、系统资源短暂不可用及系统部件之间交互失调和人为误动作等. 这些故障只会在系统运行过程中呈现, 且持续时间短、难以预测和定位, 若不加处理则会导致系统完全失效. 当前瞬时故障恢复和安全控制主要采用主动冗余和基于模型分析的故障恢复控制方法.

2.1 主动冗余

主动冗余是通过硬件冗余、软件冗余和时间冗余结合软件控制进行系统瞬时故障的检测和恢复. 下面从这 3 个方面总结使用主动冗余的故障恢复方法.

2.1.1 硬件冗余

硬件冗余是指使用硬件备份, 当瞬时故障发生时, 通过比较主部件和备份部件的输出, 选则取用正确值, 或者启用冗余替代故障部件工作, 进行故障恢复. 随着芯片集成技术的提高, 单芯片集成更多的数字单元成为可能, 更多的程序计数器、寄存器和栈的处理芯片具有线程冗余能力[11]. 现场可编程门阵列 (FPGA) 支持用户动态开发硬件电路, 通过设计电路冗余提供重构资源, 实现故障容错[12]. 在航天、电力等领域, 整个部件多模冗余运行, 启用非故障部件代替故障部件运行, 实现系统瞬时故障的恢复[13-14], 但硬件冗余一般对成本要求较高.

2.1.2 软件冗余

软件冗余相对于硬件冗余而言, 在控制成本方面具有较大优势, 并行多线程冗余和版本多样性运行是两种常见的软件冗余方法.

并行多线程冗余的基本思想[15]是在并行多线程平台上将主线程复制成多份并同时运行, 最终通过比较运行结果来屏蔽 SEUs (single event upsets) 错误. 文献 [16] 提出了一种针对多处理器芯片的 CRTR (chip-level redundantly threaded multiprocessor with recovery) 瞬时故障的检测和恢复方法, 使用基于不对称义务使能控制的长闲置部分隐藏处理器之间的延迟, 同时使用拖尾线程来恢复瞬时故障, 容忍 SEUs 错误等.

软件多样性方法是经典的软件冗余方法, 其基本思想[17]是各个版本软件由不同的团队独立设计, 使用不同的方法、不同的设计语言及不同的开发环境和工具来实现, 目的是减少各个版本软件在表决点上相关错误的概率, 藉此来实现软件系统瞬时故障的恢复. 各个版本的软件设计过程需要遵循的原则如下[18]: 总体设计相同, 避免错误恢复的全局回滚; 多样化模块之间统一接口; 多样性封装, 即模块内部多样性对外不可见; 保证各个版本软件设计独立性.

2.1.3 时间冗余

时间冗余为瞬时故障的检测恢复提供一定的执行时间, 一般结合控制软件使用. 检查点回溯恢复和软件重执行恢复是基于时间冗余结合软件进行瞬时故障恢复的常用技术.

检测点回溯是一种在故障发生情况下, 节点程序回转到最近的一个没有故障的检测点重新执行的故障恢复方法. 文献 [19] 讨论了回溯技术在时间严格受限的实时分布式系统容错中的应用. 针对数控系统对于时间确定性和可靠性的要求, 文献 [20] 基于回溯恢复机制和最佳优先算法设计了容错调度策略.

软件重执行是指在时间冗余前提下保证瞬时故障引起的故障程序可以被重新执行, 从而克服故障影响[21-23]. 文献 [21] 介绍了多标量体系结构下使用重执行进行瞬时故障容错的方法, 利用大部分处理单元执行传统的多标量计算, 将剩余的处理单元用于有效代码的重执行. 该方法能够对处理单元的瞬时故障进行有效的恢复, 但执行时间的冗余会降低系统的运行性能. [22] 针对故障的处理提出了过程重执行容错调度方法, 实际上是对重执行与静态调度算法相结合的一种优化, 降低了系统对时间冗余需求, 减少插入调度表中用于重执行的时隙数. 故障进程被重执行时, 处理器切换到一个紧急调度状态, 延迟在处理器上对该进程的执行请求. [23] 讨论了对性能和重执行透明度进行平衡的方法, 通过 FT-CPG (fault-tolerant conditional process graph) 图建立应用任务的模型, 通过调度, 既满足了系统在遇到瞬时故障时进行任务重执行, 也满足了系统性能的要求. [24] 使用基于软件控制的进程冗余方式处理多核结构系统的瞬时故障, 通过输出匹配检测、看门狗定时中断检测和程序失效来检测瞬时故障的发生, 通过周期性的检测点检测和修复技术进行故障恢复, 并通过运行进程的多备份选举正确输出进行故障掩盖, 恢复瞬时故障.

为了提高系统运行效率和降低芯片能耗, 文献 [25-26] 使用指令冗余的方法对瞬时故障进行检测和恢复. [25] 通过指令冗余进行瞬时故障检测, 针对故障的恢复提出了一种重声明的方法, 检测到瞬时故障后, 通过比较寄存器的值取用正确的值, 对执行程序的相应寄存器重新赋值, 恢复瞬时故障. [26] 将该方法用于 4 线和 8 线超标量体系结构处理器, 减少了芯片冗余, 使处理器芯片在有瞬时故障发生时芯片的性能依然有所提高.

2.2 基于模型的故障控制

基于模型的故障控制策略通过建立系统的模型, 从系统全局分析模型参数进行故障诊断, 然后制定恢复策略进行故障恢复. 目前基于系统模型故障恢复方法主要有基于解析模型和基于概念模型的方法.

基于解析模型的方法首先通过解耦建立系统的数学模型; 然后比较数学模型输出和实际测量值获取残差, 进而分析得到故障信息; 再对系统进行调节, 进行瞬时故障恢复和控制. 文献 [27-28] 全面介绍了考虑节点驱动方式、网络时延、噪声扰动、数据包丢失等进行的解析建模, 然后进行故障分析和控制的典型方法. 但由于系统的复杂性, 一般系统往往难以实现完全解耦, 时滞、建模误差和参数摄动等因素的影响使得系统的精确数学模型难以获取.

基于概念模型的方法不需要精确的数学描述, 通过建立系统的结构、操作、任务、性能等抽象模型, 设置监测获取系统的故障信息, 制定控制策略实现故障影响控制或故障恢复. 文献 [29] 提出了基于系统结构建立概念模型的方法, 通过对计算机网络的拓扑结构和连接关系进行建模, 设置约束检测系统故障, 针对节点脱离系统和加入系统引起系统瞬时故障, 从整个网络性能的角度实施调度恢复. [30] 基于结构模型构建了系统自愈框架. [31] 基于 Petri 网建立系统事件的离散行为模型, 通过状态转移关系来监测瞬时故障事件, 然后利用系统资源重分配进行故障恢复. 目前基于概念模型的瞬时故障恢复方法还不是很多, 也不够完善. 由于不依赖于精确的数学模型, 基于概念模型的方法在工程应用方面更具有优势. 另外, 运行时芯片温度升高和串扰也会导致瞬时故障发生[32], 串扰主要从电磁兼容方面进行研究, 芯片温度升高主要从芯片材料、体系结构和外部降温等进行防护, 作为电子科学方面的内容, 本文不作详述.

3 NCS 瞬时故障恢复和安全控制

NCS 规模不断扩大, 复杂程度越来越高, 网络节点和通信网络受瞬时故障影响会导致 NCS 系统失效, 引发事故. NCS 系统节点数目众多、通信网络结构复杂, 且节点均由复杂的硬件和软件构成, 系统之间交互过程也很复杂, 瞬时故障的恢复和安全控制必须全面考虑这些因素. 在 NCS 中, 瞬时故障的恢复和安全控制可以从 3 个方面实施: 网络节点瞬时故障恢复、通信网络瞬时故障恢复和系统层面瞬时故障恢复.

3.1 网络节点瞬时故障恢复和安全控制

NCS 的节点可以是执行器、嵌入式系统, 也可能是计算机控制系统. 网络节点瞬时故障可以从处理器芯片和整个节点进行恢复.

对于高能粒子等引起的处理器瞬时故障, NCS 中通常采用具有瞬时故障恢复能力的芯片容错, 或基于软件控制进行恢复. 文献 [33] 将具有 SEU 保护能力的处理器应用于太空飞行器的设计. 文献 [34] 使用容错处理器芯片结合回溯等控制软件技术进行星载计算机系统瞬时故障恢复.

网络节点本身也是一个复杂的小型数字系统, 除了高能粒子外, 外部的其他扰动和本身软件的失调都可能引起瞬时故障. 文献 [4] 总结了现代数字系统中通过冗余进行瞬时故障恢复的一些方法. [35] 针对自动控制网络系统的节点, 采用电源冗余抗击电源不稳定形成的瞬时故障, 同时通过节点完全冗余运行, 保证瞬时故障导致节点失效后启用冗余节点来使系统安全运行. 硬件完全冗余成本较大, 使用软件重执行和软件控制可以降低系统成本. [36] 讨论了当前用于分布式计算机系统节点故障处理的典型软件方法: 前向恢复和后向恢复. [37] 针对通信系统中的易变的节点提出了通过软件复制和设置检测点进行重执行的方法处理其节点瞬时故障. [38] 提出了一种轻量节点级故障容错方法, 通过使用软件控制技术对节点任务处理进行空间和时间的冗余来掩盖瞬时故障, 并将这种方法应用到线控制动系统中, 提高了系统故障处理能力和可靠性.

有一些瞬时故障不影响节点按照约定方式运行, 但会导致交互数据错误, 这种瞬时故障影响在网络节点无法检测出来, 需要从 NCS 系统层面来检测; 同时, 瞬时故障导致的失效节点通过使用节点重新初始化进行瞬时故障恢复时也需要考虑系统资源和系统全局任务调度.

3.2 通信网络瞬时故障恢复和安全控制

瞬时故障影响通信网络正常工作表现为通信短时中断、数据传输延迟和传输数据数值异变或丢包. 短时通信中断的恢复通过通信通道的硬件冗余实现, 其他方面的恢复需要在系统级进行重调度、补偿和通信重构.

文献 [35, 39-42] 利用硬件冗余讨论了通信网络短时通信中断的恢复, 主要分为全冗余和部分冗余. [35] 在自动控制网络系统中设置双通信设备结构对通信通道全冗余, 在瞬时故障的影响下, 启用冗余的对等通信线路代替故障线路进行故障恢复. [39] 针对基于 CAN 通信的控制系统, 通过对等的两条通信通道实现通信暂时中断故障的恢复. [40] 基于 IEEE802.4 的物理层, 通过通信通道冗余进行容错实现瞬时故障恢复, 同时在物理层设置控制程序来检测故障并控制通信通道切换. 通信通道的完全冗余, 可以在故障影响下使用备份完全替代以恢复系统工作,

但成本较高. [41] 通过使用一些未连接的通信节点设备备份通信信息, 当有一些节点发生瞬时故障而失效时, 激活备份节点加入通信网络来代替故障节点, 实现网络通信的瞬时故障恢复. [42] 总结了无线传感器网络中使用硬件冗余和软件冗余克服无线节点通信瞬时故障的方法. 采用部分冗余可以节省系统构建成本, 对于大型通信网络, 采用硬件冗余的方法成本很高, 只有很少的场合会使用.

对于丢包等瞬时故障, 容错通信协议是进行故障恢复的有效手段. TCP 协议[43-44]采用传输控制通过重传实现故障恢复的常用协议, 针对数据包丢失制定通过重传快速恢复的策略. CSMA 协议可以克服电磁干扰带来的数据冲突, 结合重传协议实现对于一些电磁干扰带来的物理层瞬时故障的恢复. [42] 基于时间冗余和软件控制研究了无线传感器网络中通信瞬时故障的容错方法, 通过重传实现了故障的恢复. 通信协议的重构需要从系统层面来考虑, 对于通信网络延时的补偿也需要从系统层面来实现.

3.3 系统瞬时故障恢复和安全控制

NCS 的控制器和网络节点通过通信网络进行任务交互, 对于瞬时故障导致网络延时和丢包及性能下降等需要建立系统的模型, 从整个系统性能表现进行故障诊断, 其恢复策略的制定也必须从整个网络资源出发.

通过对系统相关参数的分析、建立 NCS 系统的数学解析模型、进行故障恢复, 是 NCS 系统瞬时故障恢复的重要方式, 目前已产生较多研究成果. 文献 [28] 对其进行了部分综述. 文献 [45] 基于 DMC (dynamic matrix controller) 预测控制设计了时延补偿方法, 并将其应用于基于 Internet 的过程对象控制系统的设计: 当延时导致丢包时, 使用预测控制中的当前值作为下一次的控制值, 增强系统的稳定性. 这些研究不但对时延等故障的恢复控制进行了理论上的证明, 同时也进行了实际应用方面的研究. 分析表明, 系统数学模型的建立需要对系统运行机理和状态信息有深入的研究, 同时系统延时和摄动等动态特性的影响使得精确的系统解析模型难以获得, 工程应用受限.

对于 NCS 系统, 建立精确的瞬时故障恢复数学模型较为困难. 概念模型可以只考虑系统抽象方面建立分析模型来进行瞬时故障恢复, 易于实现和工程应用. 文献 [46] 根据系统拓扑关系建立了系统结构模型, 通过对连接关系参数的检测实现对于瞬时故障引起网络节点的退入或加入, 研究了基于系统结构模型, 从系统全局性能考虑改变调度策略或者进行网络节点功能重构以实现故障恢复. 文献 [47] 基于 NCS 的 QoS (quality of service) 建立系统通信与 QoS 的关系模型, 讨论了当瞬时故障影响系统 QoS 时, 对节点通信协议栈重构, 改变节点与网络的通信方式, 实现节点瞬时故障恢复.

从系统的层面进行瞬时故障恢复需要调度策略, 基于系统周期任务模型, 文献 [48] 介绍了分布式系统中通过静态调度实现周期任务瞬时故障容忍的方法. 但是在实际系统中, 周期任务和非周期任务往往共同存在, 因此建立了混合任务调度模型. [49] 提出 RM (rate monotonic) 和 EDF (earliest deadline first) 调度算法及其扩展算法、启发式最佳优先调度算法, 保证了对于周期任务和非周期任务的容错调度. [50] 从网络体系方面研究了瞬时故障的恢复, 提出一种 ICEBERG 网络体系结构. 通过发信号协议, 将支持多器件通信作为第 1 等级的服务, 该结构体系提供的模块和基础结构可以支持任何类型服务. 同时, 通过一种容错呼叫管理建立协议来恢复瞬时失效故障而不增加额外的逻辑负担, 也不影响系统通信的正常运行. 它为基于 Internet 的网络控制提供了一种通信瞬时故障恢复和安全控制的设计参考方案. [51] 提出基于系统理论方法的故障分析和防治方法, 从系统全局寻找防治瞬时故障发生的途径, 不再将瞬时故障当作一个单一事件, 而将其视为系统约束规则的违背. 通过分析, 建立足够的约束及监测和保护策略可防止系统瞬时故障发生, 大大提高了系统安全性和可靠性, 这种方法可以贯穿系统设计、开发、应用的整个生命周期.

目前, NCS 瞬时故障恢复从网络构成的几个重要部分进行了研究, 形成了以主动冗余和基于系统模型恢复为主的故障恢复方法. 但单一层次的瞬时故障恢复方法无法处理全部 NCS 系统的瞬时故障, 需要建立多层次瞬时故障恢复机制. 基于概念模型的瞬时故障恢复不依赖精确数学模型, 在工程实践中容易得到应用, 但目前这方面的研究还不够系统, 缺乏针对 NCS 特性的应用研究.

4 NCS 瞬时故障恢复研究展望

NCS 瞬时故障恢复和安全控制是面向技术开发的应用基础研究, 也是理论和实际相结合的研究, 因此其研究应以实际应用为导向, 理论研究与技术开发并重.

NCS 本身机构复杂, 难以获得精确的解析模型, 同时 NCS 中瞬时故障难以在单一层次进行防护, 而 NCS 资源和调度策略均会影响瞬时故障恢复的实现. 结合当前的研究现状, 以下一些方向值得进一步深入研究:

1) 基于系统外特性模型的瞬时故障检测. 传统故障检测方法都是在系统中植入代码监测系统内部性

能进行故障诊断, 随着系统控制要求增多, 系统软件代码更加复杂, 这种植入式检测方法面对多变的应用对象适应性较差. 通过轻耦合、非侵入式的监测手段, 从系统的“外部”抽象系统性能模型进行瞬时故障检测是解决这些问题的一条有效途径.

2) 多层次瞬时故障恢复和安全控制体系. NCS 结构复杂, 分布区域广, 单一层次的故障恢复处理无法满足对于整个NCS 故障恢复的要求. 建立芯片、节点和系统的多层次瞬时故障恢复和安全控制体系, 对于低层次无法处理的问题从更高层次进行控制, 是针对分布广、内部结构复杂系统瞬时故障恢复的有用尝试.

3) 多约束条件下瞬时故障的恢复策略. 瞬时故障发生具有随机性和原因难定等特征, 同时NCS 资源是受限的, 探寻在多条件约束下瞬时故障的恢复策略可以提高系统故障处理能力及可靠性.

4) 网络化环境下瞬时故障恢复策略调度. 网络化系统的故障恢复策略一般由控制器经过网络发布, 在不增加系统负担的情况下达到恢复策略的最优调度是保证系统安全性一个重要的研究课题.

5 结　论

NCS 的瞬时故障量大, 原因复杂, 对安全关键领域危害大, 处理起来比永久故障更复杂. 目前对于瞬时故障的处理已有了一些研究, 但处理能力有限, 处理结果有待进一步提高, 需要进行深入研究, 如何对NCS 瞬时故障进行有效防治和安全控制是当前迫切的任务之一.

参考文献(References)

[1] Harold E R, Brian Moriarty. System safety engineering and management[M]. New York: Wiley, 1990: 69-75.

[2] Avizienis A, Laprie J C, Randell B, et al. Basic concepts and taxonomy of dependable and secure computing[J]. IEEE Trans on Dependable and Secure Computing, 2004, 1(1): 11-33.

[3] Buja G, Menis R. Conceptual frameworks for dependability and safety of a system[C]. Int Symposium on Power, Electronics, Electrical Drives, Automation and Motion. Taormina, 2006: 44-49.

[4] Sosnowski J. Transient fault tolerance in digital systems[J]. IEEE Micro, 1994, 14(1): 24-35.

[5] Constantinescu C. Trends and challenges in VLSI circuit reliability[J]. IEEE Micro, 2003, 23(4): 14-19.

[6] 周东华, 叶银忠. 现代故障诊断与容错控制[M]. 北京: 北京大学出版社, 2000: 2-10.
(Zhou D H, Ye Y Z. Modern fault diagnosis and fault tolerant control[M]. Beijing: Peking University Press, 2000: 2-10.)

[7] 王仲生. 智能故障诊断与容错控制[M]. 西安: 西北工业大学出版社, 2005: 11-12.
(Wang Z S. Intellogent fault diagnosis and fault tolerant control[M]. Xi'an: Northwestern Polytechnical University Press, 2005: 11-12.)

[8] Li A G, Hong B R. Software implemented transient fault detection in space computer[J]. Aerospace Science and Technology, 2007, 11(3): 245-252.

[9] Mukherjee S S, Emer J, Reinhardt S K. The soft error problem: An architectural perspective[C]. Proc of the 11th Int Symposium on High-Performance Computer Architecture. Los Alamitos, 2005: 243-247.

[10] Leveson N G. System safety in computer-controlled automotive systems[C]. SAE 2000 India Mobility Conf. New Delhi, 2000: 1-8.

[11] Krishnan V, Torrellas J. A chip-multiprocessor architecture with speculative multithreading[J]. IEEE Trans on Computers, 1999, 48(9): 866-880.

[12] Nunes J L, Cunha J C, Barbosa R R, et al. Using partial dynamic FPGA reconfiguration to support real-time dependability[C]. EWDC '11 Proc of the 13th European Workshop on Dependable Computing. New York, 2011: 107-108.

[13] 邢琰, 吴宏鑫, 王晓磊, 等. 航天器故障诊断与容错控制技术综述[J]. 宇航学报, 2003, 24(3): 222-226.
(Xing Y, Wu H X, Wang X L, et al. Survey of fault diagnosis and fault-tolerance control technology for spacecraft[J]. J of Astronautics, 2003, 24(3): 222-226.)

[14] 李江, 李国庆. 容错控制在电力系统中的应用研究综述[J]. 电力系统保护与控制, 2010, 38(3): 140-146.
(Li J, Li G Q. A survey on application of fault tolerant control in power system[J]. Power System Protection and Control, 2010, 38(3): 140-146.)

[15] Vijaykumar T N, Pomeranz I, Cheng K. Transient-fault recovery using simultaneous multithreading[C]. Proc of the 29th Int Symposium on Computer Architecture. Anchorage, 2002: 87-98.

[16] Gomaa M A, Scarbrough C, Vijaykumar T N, et al. Transient-fault recovery for chip multiprocessors[J]. IEEE Micro, 2003, 23(6): 76-83.

[17] Chen L, Avizienis A. N-version programming: A fault-tolerance approach to reliability of software operation[C]. The 8th Annual Int Conf on Fault-Tolerant Computing. Toulouse, 1978: 3-9.

[18] 韩炜, 臧红伟. *N* 版本编程技术的软件可靠性分析[J]. 微电子学与计算机, 2003, 20(5): 62-63.
(Han W, Zang H W. Reliability analysis of N-version

programming[J]. Microelectronics & Computer, 2003, 20(5): 62-63.)

[19] Chandy K M, Ramamoorthy C V. Rollback and recovery strategies for computer programs[J]. IEEE Trans on Computers, 1972, 21(6): 546-556.

[20] Koo R, Toueg S. Checkpointing and rollback recovery for distributed systems[J]. IEEE Trans on Software Engineering, 1987, 13(1): 23-31.

[21] Oh N, Shirvani P P, McCluskey E J. Error detection by duplicated instructions in super-scalar processors[J]. IEEE Trans on Reliability, 2002, 51(1): 63-75.

[22] Rashid F, Saluja K K, Ramanathan P. Fault tolerance through re-execution in multiscalar architecture[C]. Proc of Int Conf on Dependable Systems and Networks. New York, 2000: 482-491.

[23] Kandasamy N, Hayes J P, Murray B T. Transparent recovery from intermittent faults in time-triggered distributed systems[J]. IEEE Trans on Computers, 2003, 52(2): 113-125.

[24] Shye A. PLR: A software approach to transient fault tolerance for multicore architectures[J]. IEEE Trans on Dependable and Secure Computing, 2009, 6(2): 135-148.

[25] Sato T, Arita I. Tolerating transient faults through an instruction reissue mechanism[C]. Proc of the ISCA 14th Int Conf Parallel and Distributed Computing Systems. Richardson, 2001: 240-247.

[26] Sato T. Exploiting instruction redundancy for transient fault tolerance[C]. Proc of the 18th IEEE Int Symposium on Defect and Fault Tolerance in VLSI Systems. Boston, 2003: 547-554.

[27] Gupta R A, Chow M Y. Networked control system: Overview and research trends[J]. IEEE Trans on Industrial Electronics, 2009, 57(7): 2527-2535.

[28] 文利燕, 彭晨, 裴灵犀. 基于模型的网络控制系统故障诊断综述[J]. 南京师范大学学报, 2011,11(1): 39- 44.
(Wen L Y, Peng C, Pei L X. Overview on fault diagnosis of networked control system based on the models[J]. J of Nanjing Normal University, 2011, 11(1): 39-44.)

[29] Garlan D, Bradley Schmerl, Cheng S W. Software architecture-based self-adaptation[J]. Autonomic Computing and Networking, 2009, 1(S): 31-55.

[30] Marija M R, Mehta Nikunj, Medvidovic Nenad. Architectural style requirements for self-healing systems[C]. Proc of the 1st ACM SIGSOFT Workshop on Self-Healing Systems. Charleston, 2002: 49-54.

[31] Odrey N G, Mejia G. A reconfigurable multi-agent system architecture for error recovery in production systems[J]. Robotics and Computer-Integrated Manufacturing, 2003, 19(1-2): 35-43.

[32] Li X B, Gaudiot J L. Tolerating radiation-induced transient faults in modern processors[J]. Int J of Parallel Programming, 2010, 38(2): 85-116.

[33] Andrea S B, Franco B. SEU protected CPU for slow control on space vehicles[C]. Proc of the 2nd IEEE Int Workshop on Electronic Design, Test and Applications. Perth, 2004: 422-424.

[34] 李剑明. 面向星载计算机瞬时故障的软件控制流错误检测技术[D]. 长沙: 国防科学技术大学研究生院, 2009.
(Li J M. Software implemented control flow error detection for transient failures in on-board computers[D]. Changsha: school of Graduate, National University of Defense Technology, 2009.)

[35] Moxa Networking Inc. Redundancy In Automation[Z]. 1301 John Reed Court, City of Industry, CA91745, 2003.

[36] 李海山. 面向恢复的容错计算技术研究[D]. 哈尔滨: 哈尔滨工程大学计算机科学与技术学院, 2007.
(Li H S. Research on recovery-oriented fault-tolerant computing technique[D]. Harbin: College of Computer Science and Technology, Harbin Engineering University, 2007.)

[37] Kanna N, Subhlok J. Redundancy tolerant communication on volatile nodes[R]. Houston: Department of Computer Science, University of Houston, 2008: 1-8.

[38] Aidemark J, Folkesson P, Karlsson J. A framework for node-level fault tolerance in distributed real-time systems[C]. Proc of 2005 Int Conf on Dependable Systems and Networks. Yokohama, 2005: 656-665.

[39] Jos'e Rufino. Redundant CAN architectures for dependable communication[R]. Lisboa: Institute Superior Tecnico, 1998: 1-14.

[40] Jae Min Lee, Wook Hyun Kwon. Physical layer redundancy method for fault-tolerant networks[C]. IEEE Int Workshop on Factory Communication Systems. Porto, 2000: 157-63.

[41] Shneidman. Using redundancy to improve robustness of distributed mechanism implementations[C]. Proc of the ACM Conf on Electronic Commerce. San Diego, 2003: 276-277.

[42] Curiac D I, Volosencu C. A redundancy and its applications in wireless sensor networks: A survey[J]. WSEAS Trans on Computers, 2009, 8(4): 705-714.

[43] RFC2582. The newReno modifieation to TCP's fast reeovery algorithm[Z].

[44] RFC2018. TCP seleetive acknowledgment options[Z].

[45] Yang S H, Chen X. Time delay and data loss compensation for internet-based process control systems[J]. Trans of the Institute of Measurement and Control, 2005, 27(2): 103-118.

(下转第1451页)