计算机科学
COMPUTER SCIENCE

基于领域分析的结构线性静力软件串并行一致化方法

唐德泓, 杨浩, 文龙飞, 徐正秋

引用本文

唐德泓, 杨浩, 文龙飞, 徐正秋. 基于领域分析的结构线性静力软件串并行一致化方法[J]. 计算机科学, 2024, 51(9): 87-95.

TANG Dehong, YANG Hao, WEN Longfei, XU Zhengqiu. Domain Analysis Based Approach to Obtain Identical Results on Varying Number of Processors for Structural Linear Static Software [J]. Computer Science, 2024, 51(9): 87-95.

相似文章推荐（请使用火狐或 IE 浏览器查看文章）

Similar articles recommended (Please use Firefox or IE to view the article)

基于主题一致性保持和伪相关反馈库扩展的缺陷报告重构方法

Bug Report Reformulation Method Based on Topic Consistency Maintenance and Pseudo-correlation Feedback Library Extension

计算机科学, 2024, 51(7): 1-9. https://doi.org/10.11896/jsjkx.230400069

基于推荐列表的缺陷文件识别

Buggy File Identification Based on Recommendation Lists

计算机科学, 2024, 51(6A): 230600088-8. https://doi.org/10.11896/jsjkx.230600088

基于跳表的secGear性能优化方法

Optimum Proposal to secGear Based on Skiplist

计算机科学, 2024, 51(6A): 230700030-5. https://doi.org/10.11896/jsjkx.230700030

残差注意力与多特征融合的图像去模糊

Image Deblurring Based on Residual Attention and Multi-feature Fusion

计算机科学, 2023, 50(1): 147-155. https://doi.org/10.11896/jsjkx.211100161

基于差分进化算法的字符对抗验证码生成方法

Adversarial Character CAPTCHA Generation Method Based on Differential Evolution Algorithm

计算机科学, 2022, 49(11A): 211100074-5. https://doi.org/10.11896/jsjkx.211100074

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231100016

基于领域分析的结构线性静力软件串并行一致化方法

唐德泓[1,2]　杨　浩[1,2]　文龙飞[1,2]　徐正秋[3]
1 中物院高性能数值模拟软件中心　北京 100088
2 北京应用物理与计算数学研究所　北京 100088
3 四川中锐信息技术有限公司　成都 610041
(tang_dehong@iapcm.ac.cn)

摘　要　并行CAE软件的计算结果串并行一致性是其计算结果可信的必要条件。然而,软件研发时常引入串并行不一致缺陷,其形式众多,现象相互耦合,散布于海量代码中,成为实现CAE软件串并行一致性的挑战。文中以结构线性静力软件的串并行一致性需求为切入点,针对现有的"专家知识法"与"缺陷定位法"应用于CAE软件串并行一致化时存在的粒度粗、准度差、成本高和缺乏系统性问题,引入领域分析方法,并与专家知识和数据流状态比对结合,提出了一种适用于结构线性静力的串并行一致化方法,实现了结构线性静力软件串并行不一致缺陷的细粒度、高准度与低成本系统性识别与修复。基于前述方法形成相关工具,并将方法与工具应用于SSTA的串并行一致化,识别并修复其中8处串并行不一致缺陷,使其通过90余真实模型的串并行一致考核,并实现串并行结果严格一致;同时,该方法与工具还将串并行不一致缺陷定位耗时由平均大于两人天降低至数人时。

关键词:串并行一致化;结构线性静力软件;串行代码并行化;领域分析;缺陷定位
中图分类号　TP311

Domain Analysis Based Approach to Obtain Identical Results on Varying Number of Processors for Structural Linear Static Software

TANG Dehong[1,2], YANG Hao[1,2], WEN Longfei[1,2] and XU Zhengqiu[3]
1 CAEP Software Center for High Performance Numerical Simulation, Beijing 100088, China
2 Institute of Applied Physics and Computational Mathematics, Beijing 100088, China
3 Sichuan Zhongrui Information Technology Co., Ltd, Chengdu 610041, China

Abstract　Obtain identical results on varying number of processors is a prerequisite for the reliability of parallel CAE software. However, during the development of parallel CAE software, various types of faults that can cause non-identical results are often introduced. These faults couple with each other to produce the final non-identical results, and are concealed within various levels of the CAE software that incorporates numerous lines of code. This poses the challenge to obtain identical simulation results on varying number of processors for parallel CAE software. When applied to parallel CAE software, traditional approaches such as expert knowledge and fault-location are often characterized by coarse granularity, poor accuracy, high cost or lack of systematism. To address this issue, we propose an approach that combines domain analysis with expert knowledge and dataflow state comparison to obtain identical results on varying number of processors for structural linear static software. This approach can be used to identify and fix faults that cause non-identical results in structural linear static software with high accuracy and low cost. Based on the above approach, we develop a corresponding tool and apply it in conjunction with the approach to identify and fix eight faults in SSTA, a structural linear static software. This endeavor helps SSTA to obtain strictly identical results on varying number of processors in more than ninety real simulation models, and significantly reduces the time required to identify a fault from more than two days to several hours.

Keywords　Obtain identical results on varying number of processors, Structural linear static software, Parallelization of serial codes, Domain analysis, Fault-location

1　引言

计算机辅助工程(下称"CAE")以数值模拟软件(下称"CAE软件")为载体,在计算机上采用数值模拟技术求解特定数学物理方程,以获得物理过程演化规律,进而解答重要装备设计、研制、生产、使用与维护中的科学技术问题,被广泛应用于航空航天、国防工业、能源工程等重大领域,已成为全面提升航空飞行器、大型舰船、装甲车辆、反应堆等重要

到稿日期:2023-10-31　返修日期:2024-04-10
通信作者:杨浩(yang_hao@iapcm.ac.cn)

装备设计、制造、使用、维护能力的关键技术手段。

随着数值模拟场景由近似简化向真实复杂、构件局部向装备整体、单尺度向跨尺度、单过程向多物理耦合的发展，其网格规模已增长至数百亿，面临内存开销极大、求解时间极长的挑战，依托并行计算机通过数百至数万核并行计算加速求解过程的并行 CAE 软件是现阶段不可或缺的技术手段。

CAE 软件串行与并行计算结果一致是其计算结果可信的必要条件[1]。从 CAE 应用角度，工程设计人员以数值模拟获得的物理过程演化规律作为设计指导，CAE 软件计算结果串并行不一致的偏差可能被误认为是物理规律，导致错误的设计决策，引发严重后果。从 CAE 软件角度，在某些数值模型与并行规模下，部分软件错误与计算机舍入误差造成的串并行结果偏差可处于同一量级，忽视串并行一致性要求，将掩盖这些错误，而在其他场景下可能产生完全失真的物理现象。

然而，实现 CAE 软件串并行一致的串并行一致化过程，需识别并修复散布于 CAE 软件多个层次的复杂算法逻辑与海量代码中的众多串并行不一致缺陷。这些缺陷形态各异，涉及多个学科且产生的串并行不一致现象相互耦合，例如数学物理方案层面依赖并行信息的数值算法与物理模型、软件并行实现层面的舍入误差与编码错误、软件并行执行层面的乱序执行与无序浮点累加归约等。这使得 CAE 软件串并行一致化并非易事。

本文关注串行代码的并行化改造[2-5]这一特定的并行 CAE 软件研制方法。在串行代码并行化过程中，数学物理方案与原程序相同，计算通信模式明确。潜藏的串并行不一致缺陷主要由重构过程中大量代码的重设计与重编写引入，集中于 CAE 软件的并行实现层面。现有串并行不一致缺陷识别方法可分为“专家知识”与“缺陷定位”两类，前者仅可粗粒度识别学科算法层面的潜在缺陷类型，后者识别成本高且易遗漏缺陷，独立使用均无法满足 CAE 软件并行实现层面串并行不一致缺陷的细粒度、高准度、低成本系统性识别与修复需求。

领域分析是软件工程学科中的一种重要方法[6]，用于建立刻画目标领域共性行为特征的领域模型，回答目标领域是什么样的问题，具有系统性、无遗漏的特点，被广泛应用于软件架构设计、测试设计等。本文以结构线性静力软件的串并行一致性需求为切入点，针对如何细粒度、高准度、低成本地系统识别与修复软件并行实现层面串并行不一致缺陷的问题，引入领域分析方法，结合专家知识与数据流状态比对技术，提出了“基于领域分析的结构线性静力软件串并行一致化方法”。首先，通过领域分析建立包含“数理方案-并行实现-并行执行”3 层技术架构、“四阶段-四组件”算法流程模型和“父子组件嵌套”层次化数据流模型的并行结构线性静力软件领域模型。其次，将前述技术架构与专家知识结合，建立了缺陷的架构层级分离方法，解耦了并行实现层面的串并行不一致缺陷。随后，将前述算法流程模型与专家知识结合，建立了串并行不一致缺陷静态定位技术，系统地识别了算法流程中各子算法的串并行不一致设计缺陷。然后，将前述层次化数据流模型与数据流状态比对方法结合，建立了串并行不一致缺陷动态捕捉技术，快速识别了持续变化且难以预测的代码实现错误。最后，将缺陷分类并针对每类缺陷提供相应的修复方法。这些方法共同应用，建立了对结构线性静力软件串并行不一致缺陷的细粒度、高准度、低成本系统性识别与修复能力。将该方法应用于结构线性静力软件 SSTA[7]的串并行一致化，识别并修复了 8 处串并行不一致缺陷，使得 SSTA 能够通过 90 余真实模型串并行一致考核，且并行实现层面串并行结果严格一致，显著提升了产品质量。同时，该方法还将串并行不一致缺陷定位耗时由平均大于两人天降低至数人时。这些应用结果表明了该方法可行且有效。

本文第 2 章剖析了 CAE 软件的串并行一致性化挑战；第 3 章详细阐述了本文提出的“基于领域分析的结构线性静力软件串并行一致化方法”；第 4 章简要介绍了基于该方法形成的工具；第 5 章详细介绍了该方法在结构线性静力软件 SSTA 串并行一致化中的应用效果；最后总结全文。

2 CAE 软件串并行一致性问题浅析

2.1 串并行一致性定义

若在任意核数并行规模下，对于相同的输入数据，并行 CAE 软件给出的结果与串行结果一致，则称该软件满足串并行一致性。

串并行一致性对于并行 CAE 软件具有重要意义。从 CAE 应用角度，数值模拟结果作为工程设计的重要依据，同一数值模型不应在不同并行规模计算中产生不同的计算结果。工程设计人员常通过调整数值模型参数进行模拟，以对比和选择不同设计方案或归纳物理规律。这些数值模型通常以不同并行规模运行，串并行不一致的偏差将被工程设计人员误判为方案或规律中的真实物理现象差异，进而导致错误的设计决策，产生严重后果。例如，在飞行器结构稳定性设计中，需要通过调整数值模型的刚度分布来归纳该分布与结构稳定性间的关系，串并行偏差可导致不同并行规模的模拟产生不同失稳现象，归纳出错误的规律，误导设计师给出错误的刚度布局，进而导致飞行器因结构失稳引发空难。

另一方面，从 CAE 软件质量角度，导致串并行模拟结果不一致的原因既可能是计算机舍入误差，也可能是并行实现错误，且两者偏差量级可相当。例如，表 1 列出了某舰船结构变形分析的最终结果位移的两种串并行偏差来源，前者由误用累加归约同步子域共享结点的法向量引起，为实现错误，后者由求和顺序变化引起，为舍入误差，两者量级相当。若未经诊断而笼统地将偏差归咎于舍入误差，则将掩盖软件中存在的错误，而此错误在其他场景下可能导致完全失真的物理现象。

表 1 某舰船结构变形分析中位移串并行差异的两种来源

Table 1 Two sources of displacement error on varying number of processors in structural deformation analysis for a ship

并行规模	误用累加归约同步共享结点法向量引起的偏差（实现错误）		求和顺序变化引起的偏差（舍入误差）	
	最大绝对误差	最大相对误差	最大绝对误差	最大相对误差
2	1.093×10^{-8}	2.896×10^{-5}	1.081×10^{-8}	2.865×10^{-5}
4	1.046×10^{-8}	2.627×10^{-5}	5.411×10^{-9}	1.490×10^{-5}
8	5.546×10^{-9}	1.357×10^{-5}	5.368×10^{-9}	6.289×10^{-6}
16	7.699×10^{-9}	1.934×10^{-5}	6.272×10^{-9}	1.722×10^{-5}
32	1.741×10^{-8}	4.579×10^{-5}	1.077×10^{-8}	2.704×10^{-5}

2.2 并行 CAE 软件的串并行一致化挑战

并行 CAE 软件的串并行一致化旨在解决“如何实现并行 CAE 软件的串并行一致性”问题，即通过技术手段，诊断并修复软件中导致计算结果串并行不一致的缺陷，使得软件串并行一致的过程。

然而，并行 CAE 软件通常采用复杂的数值算法和并行算法，涵盖计算力学、计算数学、并行计算等多个学科，可包含数十万行代码。串并行不一致缺陷可能以算法特性、舍入误差、实现错误等多种形式藏匿于不同学科的不同算法及其庞大的实现代码中，且各缺陷引发的串并行不一致现象相互耦合，使得识别和修复的任务变得困难。“如何高准度且低成本地精细识别并修复这些缺陷”是并行 CAE 软件在串并行一致化中面临的挑战。当前的串并行一致化方法主要聚焦于“缺陷识别”和“缺陷修复”两个方面。

1)串并行不一致缺陷识别方法

已知的串并行不一致缺陷识别方法主要可分为专家知识与缺陷定位两类。

(1)专家知识

该类方法利用物理、计算数学和并行计算等领域专家知识推断软件中可能存在的串并行不一致缺陷。例如，利用线性代数领域专家知识，推断采用区域分解算法可导致结果与并行规模相关。利用并行计算领域专家知识，可推断异步执行、无序浮点累加归约等可能导致结果串并行不一致。

(2)缺陷定位

该类方法是软件工程领域的研究热点[8-10]，主要分为传统定位、代码覆盖和状态监测 3 类。传统定位类方法通过从错误现场沿程序执行流或数据依赖关系反向追踪错误诱因来定位软件缺陷，以调试器追踪调试为代表。代码覆盖方法通过比较各代码段分别在正确测例与错误测例中出现的频率来识别嫌疑代码，以谱方法和概率方法为代表。状态监测方法通过比较程序在特定位置的当前状态和参考状态的一致性来隔离程序错误，以数据流状态分析和模型类方法为代表。

上述定位方法中，专家知识方法基于算法的学科抽象，只能提供潜在缺陷类别等粗粒度指导性结论，粒度、精度十分有限且缺乏系统性，无法满足 CAE 软件代码级缺陷定位需求。例如，该方法仅可推断浮点累加结果，可随计算顺序变化而改变，却无法指明程序中累加顺序依赖并行规模的代码段的位置。缺陷定位方法基于测例，识别能力高度依赖测例集对程序执行路径的覆盖程度，成本高而且难以保证其完备性，且其中各方法都有局限性。例如，并行 CAE 软件执行流、数据流高度复杂，传统定位方法的工作量和难度都很大。又如，并行 CAE 软件的测例主要源自真实应用，构造难度大、数量少，对软件复杂执行路径覆盖有限，无法满足覆盖类方法准确定位缺陷所需测试用例集规模。再如，并行 CAE 软件代码量大，为细粒度定位缺陷，通用状态监测类方法只能布设大量代码语句级检查点与状态检查变量，状态检查开销极高，而 CAE 软件持续演进又使得语句级状态检查机制面临高维护成本。状态监测类方法为了降低高精度定位的代价与自身维护成本，需结合应用领域知识进行系统性设计，但目前尚未见其在并行 CAE 软件上的实践。综上可知，独立使用现有方法离实现 CAE 软件串并行不一致缺陷的细粒度、高准度、低成本系统性识别仍有一定距离。

2)串并行不一致缺陷修复方法

已知的串并行不一致缺陷修复方法主要可分为数学物理方案修正、并行程序实现修正与并行程序执行修正 3 类。

(1)数学物理方案修正

该类方法通过将并行规模相关的数学物理算法替换为并行规模无关算法，实现了数学物理方案的串并行一致。例如，Nicholas 等[11]在其隐式蒙特卡罗粒子模拟程序中，通过将随机数生成器状态绑定至粒子，并使用区域分解无关的随机数生成算法，实现了计算结果的串并行一致。又如，Li 等[12]在三维中子-光子耦合输运 MC 软件 JMCT 中，通过赋予粒子随机数属性，并动态派生次级粒子随机数，实现了基于区域分解并行的蒙特卡罗粒子模拟的串并行一致。再如，Liu[1]在化学非平衡流模拟中采用代数多重网格法，而非依赖并行规模的区域分解法。

(2)并行程序实现修正

该类方法通过识别并修正程序中依赖并行规模信息或浮点运算顺序的算法设计与代码错误，实现了并行程序实现的串并行一致。例如，Nicholas 等[11]在其隐式蒙特卡罗粒子模拟程序中，识别出能量沉积计算包含随并行规模变化而改变顺序的浮点累加操作，并以整数替代浮点数表示能量沉积，将不具有结合律的浮点数求和变为了具有结合律的整数求和，解决了能量沉积值累加顺序变化导致的串并行不一致。

(3)并行程序执行修正

该类方法主要解决并行执行层面中乱序执行、无序浮点累加归约等导致的串并行不一致缺陷。例如，采用严格顺序的消息收发等。

3)结构线性静力软件串行代码并行化过程的特点

针对本文关注的结构线性静力软件串行代码并行化这一重要应用场景，其串并行一致化具有如下特点：(1)串并行程序数学物理方案相同，数学物理方案的变化通常会导致计算结果出现差异，为了可以使用串行代码计算结果为参照考核并行化后程序的正确性，串并行程序通常采用相同的数学物理方案。(2)使用简单的并行执行逻辑，结构线性静力软件通常都遵循“单元计算-全局累加”的计算通信模式，在并行程序执行层面主要包含影像区填充、共享数据最值归约、共享数据累加归约。其中，前两者与并行规模无关，后者可通过设置影像区将归约变为影像区以填充与局部累加的方式进行规避。(3)在复杂的并行程序实现过程中，并行化改造中包含大量代码的重设计与重编写，极易在并行程序实现层面引入串并行不一致缺陷。

本文以结构线性静力软件串行代码并行化中的串并行一致性需求为背景，研究并行程序实现层面的系统性串并行一致化技术，解决了如何细粒度、高准度、低成本地系统诊断与修复串并行不一致缺陷的问题。

3 基于领域分析的结构线性静力软件串并行一致化方法

3.1 串并行不一致缺陷分离

如前文所述，现有技术应用于结构线性静力软件并行实现层面的串并行一致化时，存在缺陷定位粒度粗、准确度不足、

成本高、缺乏系统性等问题，难以满足需求。对此，本文采用领域分析方法，结合专家知识法和数据流状态比对法，提出了基于领域分析的结构线性静力软件串并行一致化方法。该方法包括缺陷分离、缺陷诊断和缺陷修复3个步骤，能有效提高串并行一致化的定位精度、准确度与系统性，并降低成本。

为独立定位与修复结构静力软件并行实现层面的串并行不一致缺陷，首先需将其与数理方案层面、并行执行层面分离，确保串并行不一致现象互不干扰。针对此问题，基于领域分析方法，采用图1所示的领域分析流程，首先将并行结构静力软件分解为“数理方案”“并行实现”与“并行执行”3个层面，相邻层面间分别以“数值代数问题”与“并行计算原语”隔离。然后，将串并行不一致诱因分解为“算法特性”“舍入误差”与“实现错误”3类。其中，“算法特性”指算法本身依赖并行信息，与“舍入误差”及“实现错误”是否存在无关，例如区域分解法本身就依赖并行规模。“舍入误差”指计算中由计算机浮点数截断引入的差异，与“算法”本身是否依赖并行信息无关。“实现错误”指未正确实现算法意图的错误编码。上述3类诱因相互独立。最后，借助专家知识，分别从数据与算法的串并行一致性两个维度梳理每个层次内与3类串并行不一致诱因对应的差异来源。基于此梳理过程，识别得到了6种不同的串并行不一致缺陷类型。

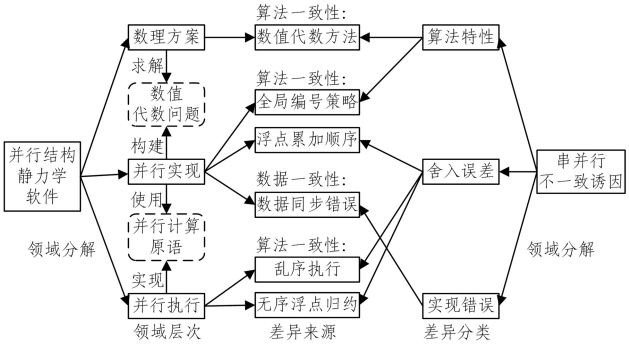

图1 并行结构线性静力软件串并行不一致缺陷领域分析

Fig. 1 Domain analysis of faults leading to non-identical results on varying number of processors in parallel structural linear static software

具体地，数理方案层面主要包括数值代数方法，用于求解结构线性静力对应的数值代数问题，例如由总体刚度矩阵、总体载荷向量与总体约束矩阵组成的鞍点问题。该层面中某些算法自身依赖网格剖分等并行信息，求解结果无法串并行一致，例如区域分解法。并行实现层面主要包括全局编号策略、浮点累加顺序与数据同步错误3种差异来源。其中，全局编号策略为网格实体、未知量以及线性系统的行列提供全局身份编号，其依赖并行信息时可导致串并行不一致，例如基于进程偏移的编号。浮点累加顺序指软件中的浮点数求和顺序，例如单元刚度元素向总体刚度元素累加的顺序，其与并行规模相关可导致串并行不一致。数据同步错误指影像数据或共享数据未正确同步或使用了依赖并行信息的同步算法，可导致串并行不一致。并行执行层面主要包括乱序执行与无序浮点归约。

上述3个层面中，数理方案与并行实现间以“数值代数问题”为界，通过检验数值代数问题的串并行一致性可解耦缺陷，如检查鞍点问题中的总体刚度矩阵、总体载荷向量和总体约束矩阵等。并行执行层面的缺陷主要包括乱序执行与无序浮点归约，前者存在成熟的串并行一致化方案，如调整编译参数，后者可通过设置影像区将累加归约变为影像区填充与局部累加的方式进行规避。通过以上方式，可将并行实现层面中的串并行不一致缺陷与其他层面解耦隔离。

3.2 串并行不一致缺陷诊断

串并行不一致缺陷诊断旨在实现缺陷的细粒度、高准度、低成本系统性定位。本文针对此目标，以前述领域分析结果为基础，提出了基于领域分析的串并行不一致缺陷诊断方法。首先，通过领域分析，建模并行结构线性静力软件，得到四阶段-四组件算法模型与父子组件嵌套层次化数据流模型。随后，针对其中潜藏的算法级与实现级两类串并行不一致缺陷，分别提出了静态识别技术与动态捕捉技术。

具体地，算法级缺陷存在于数值算法的实现方案中，主要包括“算法特性”与“舍入误差”，与算法流程密切相关，基本上不随代码实现而改变，其诊断的关键是系统性定位。对此，我们提出了串并行不一致缺陷静态识别技术，通过以专家知识分析算法流程中各子算法的串并行一致性，系统地定位潜在算法设计缺陷。代码实现缺陷指编码过程引入的错误，通常随程序研发持续出现，并由测试用例的测试结果暴露出来，形式多样，难以预测，其诊断的关键是快速定位。对此，我们提出了串并行不一致缺陷动态捕捉技术，基于数据流状态比对思想，利用结构线性静力软件数据流的层次化结构特征，将缺陷隔离在数据流的不同部分，从而显著降低缺陷定位难度，提升定位速度。

3.2.1 结构线性静力算法流程建模

结构线性静力软件通常采用有限元方法，我们基于领域分析方法对其算法流程进行建模。该建模过程以执行流与数据流为核心，以“算法阶段”与“算法组件”为分割依据，以识别主要计算、通信子算法以及数据依赖关系为目的，通过系统梳理结构线性静力并行算法，得到图2所示的包含4个阶段(场初始化、单元计算、全局组装及并行求解)、4个组件(刚度、载荷、约束及数值代数解法器)共17种子算法的算法流程模型。

具体地，场初始化阶段包含面法向量、结点法向量等辅助场的计算与全局同步。其中，面法向量通过面上结点坐标的向量运算得到，由面-点拓扑与结点坐标决定。结点法向量通过对其邻接面的法向量取均值得到，由点-面拓扑与面法向量决定。全局同步主要为共享数据同步与影像区填充。

单元计算阶段包含单元刚度矩阵、结点载荷向量与本地约束向量计算3部分。其中，单元刚度矩阵由单元形函数偏导积分导出，分为实体单元与结构单元两类，前者仅由结点坐标、单元-结点拓扑和材料参量决定，后者还与结点法向量有关。结点载荷由载荷值在所依附实体(体、面、边、结点)的形函数上积分导出，分为全局与局部两类，前者仅由结点坐标、实体(体、面、边)-结点拓扑、载荷值决定，后者因涉及载荷值投影，还与面、边等方向向量有关。本地约束向量计算可分为显式约束、绑定约束与刚性单元约束3类，显式约束直接以 $\Sigma a_i x_i = b$ 的形式定义网格结点上未知量间的关系，从而得到

本地约束向量，其中 x_i 为未知量，a_i 与 b 为系数。绑定约束通过点-面投影与形函数插值建立几何邻近网格结点上未知量间的关系，从而得到本地约束向量，由实体-结点拓扑、结点坐标决定；刚性单元约束通过计算结点坐标的投影与插值建立指定结点集中各结点上未知量间的关系，从而得到本地约束向量，由结点集中结点顺序和坐标决定。

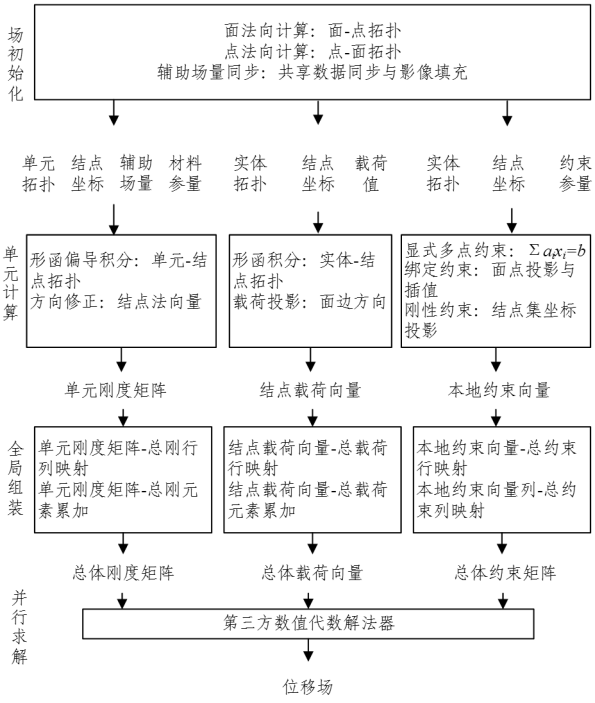

图 2 结构线性静力算法流程建模

Fig. 2 Modeling of structural linear static algorithm flow

总体组装阶段包括总体刚度矩阵、总体载荷向量与总体约束矩阵 3 部分，可分为位置映射与元素数据累加两个子过程。其中，位置映射建立单元计算阶段产生的矩阵、向量中元素到总体矩阵、向量中元素的位置映射关系，包括单元刚度矩阵行列号-总体刚度矩阵行列号、结点载荷向量行号-总体载荷向量行号、本地约束向量-总体约束矩阵行号、本地约束向量列号-总体约束矩阵列号 4 种映射关系。元素数据累加依据前述位置映射关系将单元计算阶段产生的矩阵、向量中元素累加至总体矩阵、向量对应位置，包括单元刚度矩阵-总体刚度矩阵累加以及结点载荷向量-总体载荷向量累加两部分。

并行求解阶段以总体刚度矩阵、总体载荷向量与总体约束矩阵构成鞍点问题，通过第三方数值代数解法器进行求解得到最终计算结果，即位移场。

3.2.2 结构线性静力串并行不一致缺陷静态识别

算法的串并行一致是并行软件串并行一致的前提，而如何系统性地识别算法串并行不一致缺陷是诊断面临的关键问题。对此，我们提出了表 2 所列结构线性静力串并行不一致缺陷静态识别技术。该技术基于前述“四阶段-四组件”算法流程模型，从算法是否依赖并行信息、是否依赖计算顺序和并行信息与计算顺序是否串并行一致 3 个角度，根据专家知识逐一分析算法流程模型中 17 种子算法的串并行一致性，共识别出 8 处缺陷。该技术基于领域分析引入的算法流程模型，确保了缺陷识别的系统性，并将专家知识法的缺陷识别精度由潜在缺陷类别等粗粒度的指导性结论提升至细粒度的子算法。

表 2 串并行不一致缺陷静态识别结果

Table 2 Static identification of faults leading to non-identical results on varying number of processors

计算阶段	组成部分	子算法	是否依赖并行信息	是否存在计算顺序依赖	串并行是否一致
场初始化		面法向量计算	否	依赖面-结点拓扑顺序	是
		点法向量计算	否	依赖结点-面拓扑顺序	否
		影像区填充	否	否	是
		共享数据同步	否	否	是
单元计算	单元刚度矩阵计算	形函数导数积分	否	依赖单元-结点拓扑顺序	是
		基于几何方向修正	否	否	是
	结点载荷向量计算	载荷值投影	否	否	是
		形函数积分	否	依赖体/面/边-结点顺序	是
	本地约束向量计算	显式多点约束计算	否	否	是
		绑定约束计算	否	否	是
		刚性单元约束计算	否	依赖结点顺序	否
全局组装	总体刚度矩阵组装	单元刚度矩阵-总刚阵行列映射	是	否	否
		单元刚度矩阵-总刚阵元素累加	否	依赖结点-单元拓扑顺序	否
	总体载荷向量组装	结点载荷向量行-总载荷行映射	是	否	否
		结点载荷乡里-总载荷元素累加	否	依赖结点-体/面/边拓扑顺序	否
	总体约束矩阵组装	本地约束向量-总约束行映射	是	否	否
		本地约束向量列-总约束列映射	否	否	否

3.2.3 结构线性静力串并行不一致缺陷动态捕捉

实现错误种类多且随软件研发持续增加，散布于结构静力软件中，由测试暴露出来，如何快速定位是诊断面临的关键问题。对此，我们基于数据流状态比对思想，提出结构线性静力串并行不一致缺陷动态捕捉技术，实现错误的自动隔离，从而加速错误定位。

数据流状态比对是一种重要的状态监测类软件缺陷定位思想，其通过在程序数据流中不同位置设置多个检查点，并检验各检查点上特定变量集取值与参考数据是否一致来判定检查点间代码是否存在错误，从而自动隔离缺陷，缩小调试搜索范围。然而，该思想能否成功落地取决于能否设计数量可控且稳定的细粒度检查点与检查变量集。

基于前述结构线性静力算法流程领域的分析结果，我们发现并行结构线性静力软件的数据流符合图 3 所示的示父子

组件嵌套的层次化数据流模型。

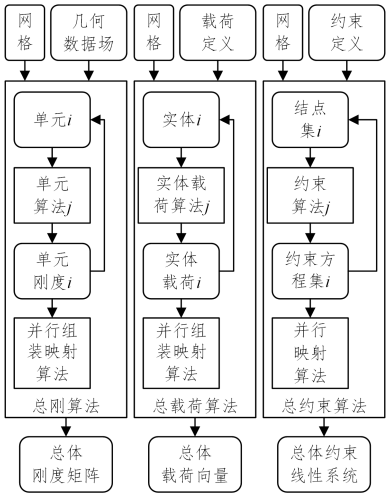

图 3 结构线性静力软件层次化数据流模型

Fig. 3 Hierarchical data flow model of structural linear static algorithm

此模型中，子组件数据流是父组件数据流的一部分，其状态变化由父组件提供的接口读参数与自身算法逻辑决定，且其仅可通过修改接口写参数改变父组件数据流来对外部状态产生影响。父子组件间的接口能有效隔离程序数据流的状态空间，为基于数据流状态的比对提供了天然的检查点与有限且完备的变量集。

基于此，我们提出了结构线性静力串并行不一致缺陷动态捕捉技术。该技术以前述算法组件的进入接口与返回接口为检查点，以接口读参数与写参数为检查变量集，以程序串行执行输出的接口参数值为参考数据。程序并行执行时，通过在组件接口自动读入参考数据，并与当前数据比对，即可实现缺陷自动隔离。

具体地，该技术将流经组件的数据流分割为上游、组件中与下游 3 段，从而将错误隔离至对应代码段或外部输入中。例如，总刚计算中，分别以总刚算法组件和单刚算法组件的进入接口与返回接口为检查点，以网格、几何数据场、单元-结点拓扑、结点坐标等读参数为入口检查变量，以总刚、单刚等写参数为出口检查变量。则如图 4 所示，流经总刚组件的数据流被分割为总刚组件上游、单刚组件上游、单刚组件中、单刚组件下游、总刚组件下游 5 段。

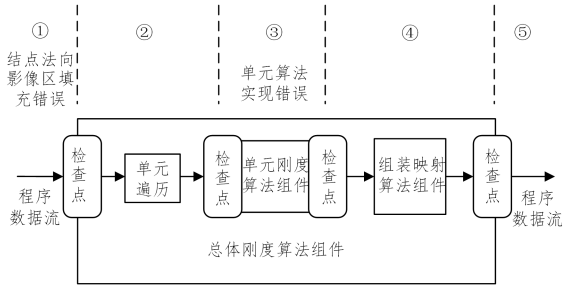

图 4 总体刚度算法组件缺陷隔离

Fig. 4 Fault isolation in the algorithm component of global stiffness matrix

基于此，未填充结点法向量场影像区导致的错误被隔离在总刚组件上游，由总刚组件或单刚组件的进入接口检查点捕获，而单刚组件内部实现错误被隔离在单刚组件中，由单刚组件的返回接口检查点捕获。

由于该技术基于前述“父子组件嵌套”层次化数据流模型，只需组件个数量级的有限数量检查点与组件接口参数个数量级的极小化检查变量集即可实现细粒度的错误自动隔离。同时，因前述数据流模型与并行结构线性静力软件架构基本一致，并不随新组件研发、旧组件重构而频繁变化。相比采用代码语句级检查的通用数据流状态比对法，本文方法可有效降低状态比对开销和自身维护成本。

3.3 串并行不一致缺陷修复

基于前述诊断的结果，我们将识别出的结构线性静力软件并行实现层面的串并行不一致缺陷分为表 3 所列的算法特性、舍入误差、实现错误 3 类，共包含“全局编号串并行不一致”“局部计算顺序串并行不一致”“共享数据同步错误”“影像区数据未同步”4 种缺陷，并为各种缺陷提供了相应的修复方法。例如，全局编号串并行不一致缺陷通过构造并行规模无关的编号算法来修复。又如，局部计算顺序串并行不一致缺陷通过构造并行规模无关的计算顺序来修复。

表 3 串并行不一致缺陷类型、实例与修复方法

Table 3 Types, examples and repair approaches of faults leading to non-identical results on varying number of processors

类型	缺陷	缺陷实例	修复方法
算法特性	全局编号串并行不一致	(1)单刚-总刚行列映射采用基于进程的偏移排序 (2)实体载荷-总载荷行映射采用基于进程的偏移排序 (3)本地约束-总约束行映射采用基于进程的偏移排序 (4)本地约束列-总约束列映射采用基于进程的偏移排序	构造并行规模无关编号算法：例如以并行无关的〈结点全局编号，未知量编号〉作为矩阵、向量元素身份标识，基于稳定排序算法建立映射关系
舍入误差	局部计算顺序串并行不一致	(1)单刚-总刚元素累加 (2)实体载荷-总载荷元素累加 (3)邻接面法向向量向壳单元结点法向向量的累加 (4)刚性单元约束计算顺序受共享结点数据收集时的无序消息接收顺序影响	构造并行规模无关计算顺序：例如依据元素所依附网格实体全局 ID 顺序
实现错误	共享数据同步错误	(1)以错误的归约操作类型同步共享实体上的数据	修改归约操作类型
	影像区数据未同步	(1)未填充结点法向数据影像区，导致影像区壳单元刚度串并行不一致	增加影像区填充过程

4 层次化数据流状态自动比对工具

为实现前述“结构线性静力串并行不一致缺陷动态捕捉”，我们研制了由图 5 与图 6 组成的层次化数据流状态自动比对工具。

其中，图 5 定义了各算法组件读写参数组成的检查

变量集；图 6 所示为通过在各算法组件前后分别增加“读参数检查点”与“写参数检查点”形成“算法组件检查代理”，该代理基于参考数据检查组件输入与输出的串并行一致性。

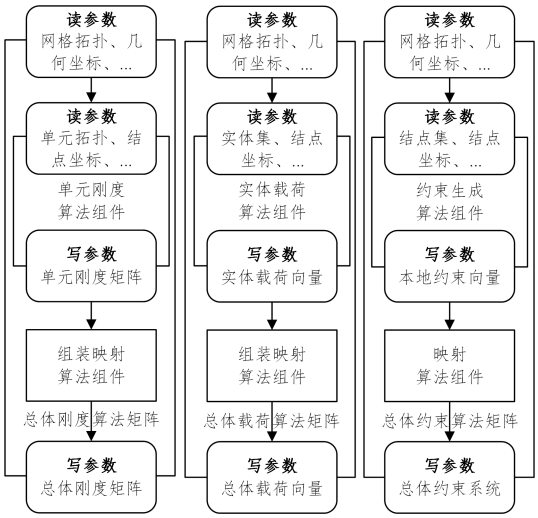

图 5 算法组件检查变量集

Fig. 5 Check variable set of algorithm components

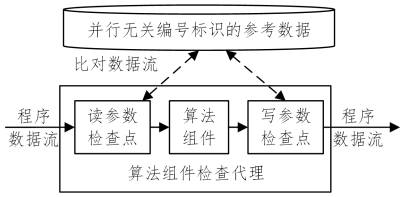

图 6 算法组件代理

Fig. 6 Agent of algorithm components

具体地，检查点定义刻画当前数据流状态的变量集合与变量数据的 IO 格式，例如壳单元刚度算法组件的读参数检查点包含结点坐标、结点法向量与材料参数 3 组变量，并采用每个单元一个文本文件的 IO 格式。同时，检查点提供了输出与比对两种工作模式，并可通过环境变量进行切换，前者用于产生参考数据，后者用于加载参考数据、完成比对及输出比对结果。此外，比对模式还支持自定义比对准则、结果内容和格式。例如，单元刚度算法检查点以绝对误差作为比对准则，以比对失败的单元全局 ID、变量名、变量绝对误差值等作为比对结果。

该工具通过在输出模式下串行执行软件生成参考数据，并在比对模式下以不同并行规模执行软件完成自动比对。上述过程中，参考数据需在不同并行规模间传递，由于程序串并行执行的指令流不同，传统以指令流中序号标识参考数据身份的方式并不适用于此处的串并行比对。为此，我们利用结构线性静力软件中算法组件的各次调用与网格中不同实体、实体集对应的特点，以并行规模无关的实体、实体集全局 ID 来标识算法组件各次调用输入、输出的数据身份。例如，以单元全局 ID 标识单元刚度算法组件各次调用的输入、输出数据。

5 应用实例

5.1 整体应用效果

本文将上述"基于领域分析的结构线性静力软件串并行一致化方法"应用于基于非结构网格并行编程框架 JAUMIN[13] 的并行结构线性静力软件 SSTA[7] 的并行化。通过该方法，识别并修复了表 4 所列的 8 处串并行不一致缺陷，使得 SSTA 能够通过卫星、舰船、飞机等领域 90 余个真实数值模型的串并行一致性考核，并在并行实现层面上保证串并行计算结果严格一致。这一改进显著提升了 SSTA 的软件质量。

表 4 SSTA 串并行不一致缺陷

Table 4 Faults leading to non-identical results on varying number of processors in SSTA

编号	缺陷
1	本地约束向量列号到总体约束矩阵列号映射采用基于进程的偏移排序
2	本地约束向量到总体约束矩阵行号映射采用基于进程的偏移排序
3	本地单元顺序变化导致单元刚度矩阵元素到总体刚度矩阵元素的累加顺序变化
4	本地面、边、结点等实体顺序变化导致实体载荷向量元素到总体载荷向量元素的累加顺序变化
5	因本地结点-面拓扑变化导致壳单元结点法向计算中邻接面法向向量向结点法向向量的累加顺序变化
6	刚性单元约束计算顺序受共享结点数据收集时的无序消息接收顺序影响
7	壳单元结点法向量场未填充影像区，导致影像区壳单元刚度计算串并行不一致
8	误以累加归约方式同步结点法向量数据场在子域边界共享结点上数据，导致其法向量错误

同时，在 SSTA 软件的并行化过程中，我们将"层次化数据流状态自动比对工具"嵌入 SSTA 中。该工具的使用将错误定位成本从平均超过两人天降低到数人时，有效地节省了人力。以下是具体的实例。

5.2 SSTA 舰船仿真串并行不一致缺陷定位

采用 SSTA 对某舰船进行结构变形分析的数值模型如图 7 所示。该模型的计算域包含上部人字形结构和下部曲面结构两部分，分别采用四面体单元和四边形壳单元进行离散，并使用相同线弹性材料，单元总数为 79354。同时，曲面结构的 4 条边被施加位移约束，人字形结构的顶部管内壁和一侧脚底部被施加结点载荷。

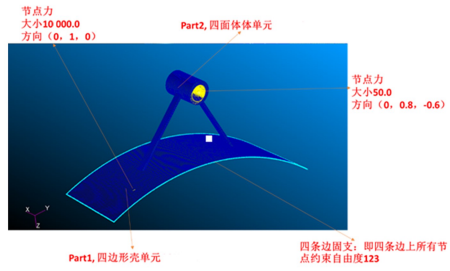

图 7 某舰船结构变形分析数值模型

Fig. 7 Numerical model of a ship for structural deformation analysis

以总体刚度矩阵、总体载荷向量与总体约束矩阵中元素在串行执行与并行执行时取值的最大绝对偏差与最大相对偏差为评价指标，考核 SSTA 软件在 4，8，16，32 核并行规模下的串并行一致性。考核结果表明，总体载荷向量与总体约束矩阵在所有规模下均串并行严格一致。然而，如图 8 所示，总体刚度矩阵在 4，8，16，32 核并行规模下分别有 3024，11 802，14 783，25 008 个元素（error_item_count）的取值存在串并行

差异，最大绝对误差(max_absolute_error)分别为 1.12×10^{-8}，1.12×10^{-8}，1.49×10^{-8}，1.04×10^{-7}，最大相对误差(max_relative_error)分别为 1.60×10^{-3}，8.07×10^{-12}，2.69×10^{-11}，2.69×10^{-11}。


```
"process": 4,
"max_absolute_error": 1.1175870895385742e-08,
"max_relative_error": 0.0015600624024961,
"error_item_count": 3024,
"size": 8738235

"process": 8,
"max_absolute_error": 1.1175870895385742e-08,
"max_relative_error": 8.067969200235286e-12,
"error_item_count": 11802,
"size": 8738235

"process": 16,
"max_absolute_error": 1.4901161193847656e-08,
"max_relative_error": 2.6904729322627573e-11,
"error_item_count": 14783,
"size": 8738235

"process": 32,
"max_absolute_error": 1.043081283569336e-07,
"max_relative_error": 2.6904729322627573e-11,
"error_item_count": 25008,
"size": 8738235
```


图 8 缺陷修复前的总体刚度矩阵串并行误差评价

Fig. 8 Error evaluation of global stiffness matrix on varying number of processors before restoration

为了快速定位该差异的来源，采用前述层次化数据流自动比对工具，按照图 9 所示的流程定位串并行不一致缺陷。首先，通过环境变量开启工具的输出模式，以单核执行程序，获得参考数据文件。

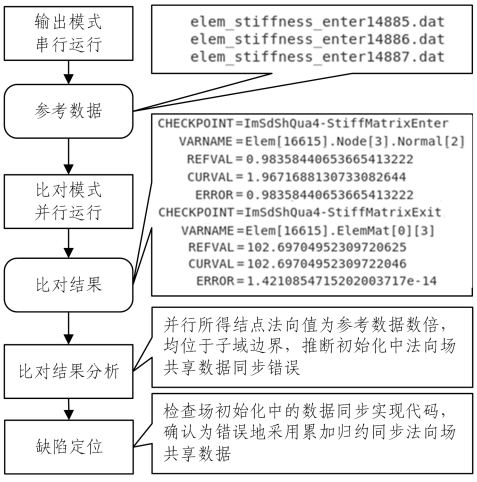

图 9 缺陷定位工作流

Fig. 9 Workflow to identify faults

随后，通过环境变量开启工具的比对模式，分别以各并行规模运行程序，获得自动比对结果。该结果给出了比对失败的检查点名、变量名与变量的参考值、当前值以及误差。以图 9 所示的 8 核的部分比对结果为例，ImSdShQua4StiffMatrix 单元算法组件在处理 16615 号单元时，输入参数(Enter)中该单元 3 号结点的法向(Normal)并行不一致，表明缺陷可能位于该组件上游的结点法向计算部分。

然后，结合区域分解特点分析比对结果发现，并行计算所得结点法向值(CUR_VAL)恰为参考数据(REF_VAL)的数倍，且存在偏差的结点均位于子域边界，由此可以推断存在共享数据同步错误。

最后，通过检查场初始化代码中的数据同步实现，确认缺陷为共享数据同步错误地采用了累加归约。在将其修改为差异检查归约后，总体刚度矩阵、总体载荷向量与总体约束矩阵均串并行严格一致。其中，总体刚度矩阵串并行一致考核结果如图 10 所示。


```
"process": 4,
"max_absolute_error": 0.0,
"max_relative_error": 0.0,
"error_item_count": 0,
"size": 8738235

"process": 8,
"max_absolute_error": 0.0,
"max_relative_error": 0.0,
"error_item_count": 0,
"size": 8738235

"process": 16,
"max_absolute_error": 0.0,
"max_relative_error": 0.0,
"error_item_count": 0,
"size": 8738235

"process": 32,
"max_absolute_error": 0.0,
"max_relative_error": 0.0,
"error_item_count": 0,
"size": 8738235
```


图 10 缺陷修复后的总体刚度矩阵串并行误差评价

Fig. 10 Error evaluation of global stiffness matrix on varying number of processors after restoration

上述缺陷定位过程中，借助层次化数据流比对工具，很快将错误来源锁定至壳单元结点法向量场串并行不一致。这样省去了对四面体单元与壳单元两种单元刚度算法组件在各网格单元上数万次调用大量单元刚度计算代码和总体刚度矩阵组装代码的排查，有效降低了缺陷追踪耗时，仅用 1 人时即完成了缺陷的定位与修复。

结束语 以结构线性静力软件串并行一致性需求为切入点，本文针对现有“专家知识法”“缺陷定位法”识别该领域串并行不一致缺陷时面临的粒度粗、准度差、成本高、缺乏系统性的问题，将领域分析技术与专家知识法和数据流状态比对法结合，并加以对缺陷的分类修复方法，提出了一种基于领域分析的结构线性静力软件串并行一致化方法，并形成了相关工具，实现了对结构线性静力软件并行实现层面中串并行不一致缺陷的细粒度、高准度、低成本系统性识别与修复。将该方法应用于 SSTA 的并行化，识别并修复了 8 处串并行不一致缺陷，使得 SSTA 能够通过 90 余真实模型的串并行一致性考核，并在并行实现层面上保证串并行计算结果严格一致。同时，该方法还将串并行不一致缺陷定位成本由平均大于两人天降低至数人时。这些应用结果证明了该方法的可行性和有效性。

本文方法通过领域分析形成了“四阶段-四组件”算法流程模型，该模型将结构线性静力软件拆解为 17 种子算法，结合专家知识逐子算法分析串并行一致性，即实现了对软件中串并行不一致算法缺陷的识别，相比通用专家知识法，既增加了缺陷识别的系统性，又将缺陷定位精度由缺陷类型等粗粒度的指导性结论提升至细粒度的子算法。同时，该方法还通过领域分析形成了“父子组件嵌套”层次化数据流模型，该模型中的组件接口隔离了程序状态，为数据流状态比对提供了匹配软件架构的组件数量个检查点与组件接口参数数量个检查变量，结合数据流比对法，即实现了对软件中串并行不一致实现错误的自动隔离，相比采用代码语句级检查的通用数据流状态比对方法，避免了大量语句级检查点与检查变量带来

的海量状态比对高开销和软件代码持续演进带来的比对机制自身高维护成本。

然而，目前上述方法在结构线性静力软件串并行一致化方面仍存在以下不足：1）精度类缺陷的根因有待进一步研究，以形成对精度类缺陷的统一修复方案，如明确全局编号顺序变化等算法特性类缺陷引起的差异是否最终可归咎于累加顺序变化导致的舍入误差等；2）动态捕捉中由检查点检查结果推断具体缺陷位置的过程较初级，结合学科领域语义与静态分析技术识别数据依赖关系，以自动推断缺陷位置的方法有待研究；3）未将数理方案与并行执行纳入研究范畴，仅保证了并行实现的串并行严格一致，离完全解决串并行一致化的目标仍有一定距离。

由于领域分析是一种普适的软件工程方法，本文基于领域分析的串并行一致化思路可推广至其他数值模拟学科领域，例如流体、电磁等。本文形成的缺陷定位与修复方法可复用于以结构线性静力学为基础的非线性静力学、接触静力学、屈曲分析等静力学软件，其中部分技术还可复用于冲击动力学等其他力学软件。

参 考 文 献

[1] LIU X. Research on CFD Parallel Computing Technology and Massively Parallel Computing Platform for Chemical Non-equilibrium Flow Problems[D]. Zhengzhou: PLA Information Engineering University, 2006.

[2] POOYAN D, RICCARDO R, MARISA G, et al. Migrations of a Generic Multi-Physics Framework to HPC Environments [J]. Computers & Fluids, 2013, 80(2013): 301-309.

[3] FU Y G, WANG X, FENG J C. Parallel Refactor of KYLIN-II base on JCOGIN Framework[R]. Beijing: CAEP Software Center for High Performance Numerical Simulation, 2019.

[4] NATHALIE M. Industrial Code Modernization of High Performance Computing Simulations on Modern Supercomputer Architectures[D]. Paris: Paris-Saclay University, 2019.

[5] JIANG S L, XU K L, YU Y, et al. Migration of Application Software to JAUMIN/JASMIN Framework[R]. Beijing: Technical Report of CAEP Software Center for High Performance Nume-rical Simulation, 2019.

[6] IRIS R B, AMON S, TONY C, et al. Domain Engineering[M]. Berlin: Springer, 2013.

[7] WEN L F, WANG J T, ZHANG A Q, et al. Design of Structural Mechanics Solver Library[R]. Beijing: CAEP Software Center for High Performance Numerical Simulation, 2020.

[8] WONG W E, GAO R Z, LI Y H, et al. A Survey on Software Fault Localization[J]. IEEE Transactions on Software Engineering, 2016, 42(8): 707-740.

[9] PRIYA P, MIRAL P. Software Fault Localization: A Survey[J]. International Journal of Computer Applications, 2016, 154(9): 6-13.

[10] JOSEP S. A Survey on Algorithmic Debugging Strategies[J]. Advances in Engineering Software, 2011, 42(11): 976-991.

[11] NICHOLAS G, MALVIN K, THOMAS B. Obtaining identical results on varying numbers of processors in domain decomposed particle monte carlo simulations [R]. UCRL-PROC-210823, Lawrence Livermore National Laboratory, 2005.

[12] LI G, ZHANG B Y, DENG L, et al. Application of Seudo-Random Number to Obtain Identical Results on Varying Numbers of Processors in Domain Decomposed Particle Monte Carlo Simulations[J]. Chinese Journal of Computational Physics, 2017, 34(1): 67-72.

[13] LIU Q K, MO Z Y, ZHANG A Q, et al. A Programming Framework for Large-Scale Numerical Simulations on Unstructured Meshes[J]. CCF Transactions on High Performance Computing, 2019, 1: 35-48.

TANG Dehong, born in 1988, Ph.D, associate researcher. His main research interests include CAE, parallel computing framework and HPC.

YANG Hao, born in 1990, Ph.D, assistant professor. His main research interests include CAE, parallel computing framework and HPC.

（责任编辑：喻藜）